TÜRK-ALMAN ÜNİVERSİTESİ
TÜRKISCH-DEUTSCHE UNIVERSITÄT

T.C.
TURKISH-GERMAN UNIVERSITY
INSTITUTE OF THE GRADUATE STUDIES
IN SCIENCE AND ENGINEERING

EXPLANATORY COMPARATIVE STUDY OF AI MODELS IN FACE EXPRESSION RECOGNITION

Master's Thesis

Fulya YENİLMEZ

ISTANBUL 2023

TÜRK-ALMAN ÜNİVERSİTESİ
TÜRKISCH-DEUTSCHE UNIVERSITÄT

T.C.
TURKISH-GERMAN UNIVERSITY
INSTITUTE OF THE GRADUATE STUDIES
IN SCIENCE AND ENGINEERING

EXPLANATORY COMPARATIVE STUDY OF AI MODELS IN FACE EXPRESSION RECOGNITION

Master's Thesis

Fulya YENİLMEZ

M.Sc., Robotics and Intelligent Systems Turkish-German University, 2023

Advisor
Prof. Dr. Mukden UĞUR

Submitted to the Institute of the Graduate Studies in
Science and Engineering in partial fulfillment of the requirements for the
Master's degree

ISTANBUL 2023

EXPLANATORY COMPARATIVE STUDY OF AI MODELS

IN FACE EXPRESSION RECOGNITION

APPROVED BY:

Prof. Dr. Mukden UĞUR .....
(Thesis Advisor)

Prof. Dr. Ali Gökhan YAVUZ .....
Turkish-German University

Assoc. Prof. Dr. Göksel BİRİCİK .....
Yıldız Technical University

DATE OF APPROVAL: 21 July 2023

DECLARATION OF AUTHENTICITY

I declare that I completed the master thesis independently and used only the materials that are listed. All materials used, from published as well as unpublished sources, whether directly quoted or paraphrased, are duly reported. Furthermore, I declare that the master's thesis, or any abridgment of it, was not used for any other degree-seeking purpose and give the publication rights of the thesis to the Institute of the Graduate Studies in Science and Engineering, Turkish-German University.

Signature

Fulya YENİLMEZ

ABSTRACT

EXPLANATORY COMPARATIVE STUDY OF AI MODELS IN FACE EXPRESSION RECOGNITION

ISTANBUL 2023, 68 pages

This thesis provides an analysis and comparison of five widely used CNN architectures in face expression recognition task. The objective is to evaluate the performance of different CNN architectures available in the Keras library for this challenging computer vision task. The chosen CNN architectures for comparison include VGG19, InceptionV3, ResNet152V2, MobileNetV2, and EfficientNetV2B1. Two different kinds of facial datasets are used in this research. The first dataset is Fer2013, a commonly used dataset in this domain known for its unbalanced structure. The second dataset is the FACES dataset from the Max-Planck Institute, comprising posed images of individuals with a balanced structure. Both of these datasets contain labeled face expression images. Pre-processing steps, such as rotation, shift, resizing, and rescaling, are applied to these images. Since this is a comparative analysis study, the same transfer learning steps are applied to all models. The results from this training are evaluated using test accuracy, which is necessary for analyzing every aspect of the study. To ensure a fair comparison, the same transfer learning steps are applied to all models. The models are trained by dividing the dataset into three sets; training data set, validation data set, and test data set.

The comparative study reveals that each CNN architecture exhibits different levels of performance in facial expression recognition tasks. The study provides significant knowledge about the strengths and weaknesses of each CNN architecture in face expression recognition.

Overall, this comparison study provides clarity on the effectiveness of VGG19, InceptionV3, ResNet152V2, MobileNetv2, and EfficientNetV2B1 architectures in recognizing facial expressions. It serves as a valuable resource for researchers in the fields of computer vision and expression analysis.

Keywords: *Face Expression Recognition, Neural Networks, CNN Architectures, Transfer Learning, Fine-tuning*

ÖZET

YÜZ İFADESİ TANIMA ALANINDA YAPAY ZEKA MODELLERİNİN KARŞILAŞTIRILMALI AÇIKLAMALI ÇALIŞMASI

2023 , 68 sayfa

Bu tez, yüz ifadesi tanıma görevinde yaygın olarak kullanılan beş CNN mimarisinin analizini ve karşılaştırmasını sunmaktadır. Amaç, bu zorlu bilgisayarla görme görevi için Keras kütüphanesinde bulunan farklı CNN mimarilerinin performansını değerlendirmektir. Karşılaştırma için seçilen CNN mimarileri arasında VGG19, InceptionV3, ResNet152V2, MobileNetV2 ve EfficientNetV2B1 bulunmaktadır.

Bu araştırmada iki farklı türde yüz veri seti kullanılmıştır. İlk veri seti, dengesiz yapısıyla bilinen ve bu alanda yaygın olarak kullanılan Fer2013 veri setidir. İkinci veri seti ise Max-Planck Enstitüsü'nün FACES veri setidir ve dengeli bir yapıya sahip bireylerin pozlanmış görüntülerinden oluşmaktadır. Bu veri setlerinin her ikisi de etiketli yüz ifadesi içeren görüntülerdir. Bu görüntülere döndürme, kaydırma, yeniden boyutlandırma ve yeniden ölçeklendirme gibi ön işleme adımları uygulanmıştır. Bu bir karşılaştırmalı analiz çalışması olduğu için tüm modellere aynı transfer öğrenme adımları uygulanır. Bu eğitimden elde edilen sonuçlar, çalışmanın her yönünü analiz etmek için gerekli olan test doğruluğu kullanılarak değerlendirilir. Adil bir karşılaştırma sağlamak için tüm modellere aynı transfer öğrenme adımları uygulanmıştır. Modeller, veri seti üç kümeye bölünerek eğitilir; eğitim veri seti, doğrulama veri seti ve test veri seti.

Karşılaştırmalı çalışma, her CNN mimarisinin yüz ifadesi tanıma görevlerinde farklı performans seviyeleri sergilediğini ortaya koymaktadır. Çalışma, yüz ifadesi tanı-

mada her bir CNN mimarisinin güçlü ve zayıf yönleri hakkında önemli bilgiler sağlamaktadır.
Genel olarak, bu karşılaştırma çalışması VGG19, InceptionV3, ResNet152V2, MobileNetv2 ve EfficientNetV2B1 mimarilerinin yüz ifadelerini tanımadaki etkinliği hakkında netlik sağlamaktadır. Bilgisayarla görme ve ifade analizi alanlarındaki araştırmacılar için değerli bir kaynak niteliğindedir.

Anahtar Kelimeler: *Yüz İfadesi Tanıma, Sinir Ağları, CNN Mimarileri, Öğrenme aktarımı, İnce ayarlama*

ACKNOWLEDGMENTS

I would like to thank my supervisor, Prof.Dr. Mukden Uğur, for all the guidance, support, and inspiration throughout the thesis process. His critical and objective thinking was my guide to success.

I would like to express my deepest gratitude to Dr.Canan Yıldız for her mentorship, valuable insights, and patience during my thesis. Dr. Yıldız's expertise in Neural Networks has made significant contributions to the quality of this thesis. This thesis would not have been possible without her insightful feedback and constant encouragement.

I would like to thank Prof.Şeniz Ertuğrul, Prof.Ali Gökhan Yavuz, and Assoc. Prof. Kaya Oğuz for their support and guidance in my academic journey.

I also want to express my gratitude to the Computer Engineering department for allowing me to occupy their server consistently.

I also thank my friends for always being there for me.

Lastly, I would like to express my deepest gratitude to my grandparents Semiha and Ahmet, for their love and support. Being in this family has been the greatest privilege of my life, and I will forever be grateful for their influence on me into the person I am today; thank you from the bottom of my heart.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

FER	Face Expression Recognition
CNN	Convolutional Neural Network
NN	Neural Networks
FFNN	Feed-Forward Neural Networks
SGD	Stochastic Gradient Descent
ADAM	Adaptive Moment Estimation
GAP	Global Average Pooling
VGG	Visual Geometry Group
ReLU	Rectified Linear Unit

CHAPTER 1

INTRODUCTION

The face expression recognition task is one of the popular tasks in the field of computer vision. Face expression recognition is an important subject to develop human-computer interaction in robotics, especially in social robotics[6].

Achieving the highest accuracy in face expression recognition is of the most importance for effective implementation in the field of human-computer interaction. To attain the best accuracy, careful consideration of both the dataset selection and the neural network's structure is essential. These factors play a vital role in optimizing the performance of the recognition task and ensuring its successful integration into various applications within the human-computer interaction domain.

The Fer2013 dataset is a widely preferred dataset in face expression recognition. It contains over 35,000 facial images that are categorized into seven emotion classes; anger, disgust, fear, happiness, sadness, surprise, and neutral. The Fer2013 dataset is a dataset composed of images that are publicly available. These publicly available images caused the dataset being widely used in expression recognition.

The study also utilizes the FACES dataset collected by the Max-Planck Institute for Biological Cybernetics. Unlike the Fer2013 dataset, this dataset includes posed images with six expressions: neutrality, sadness, disgust, fear, anger, and happiness. The FACES dataset is not publicly accessible without permission and provides a balanced distribution of data.

In recent years, the use of deep learning architectures has shown exceptional accomplishments in various computer vision assignments, including the recognition of facial expressions. The Keras library helped to make the implementation process of deep

learning models easy[5]. As a result, a remarkable achievement in the face expression recognition task is achieved.

This thesis performs an extensive analysis and comparison of the most commonly used CNN architectures in face expression recognition using both Fer2013 and FACES datasets. The aim is to gain insights into the performance of these architectures in this specific task, considering the differences between the two datasets.

1.1 Problem and importance of problem

Face expression recognition is one of the most challenging tasks in the computer vision field. Some of the reasons this task is challenging; the variety of human facial expressions and the confusion between expressions. It is challenging to precisely identify and distinguish between facial expressions due to the wide range of expressions.

There is more behind this challenging part, such as various factors in the image. These factors include the lighting condition provided in the image. In most images, the light on the faces can be too bright or too dark, making it harder to detect facial expressions. Other significant factors are glasses, facial hair on the face, and individual differences in face shape.

In numerous applications, it is vital to perform face expression recognition. One field that uses facial expression is the field of human-computer interaction. Recognizing human expressions can allow computers and intelligent machines to respond or act according to them. When the computer responds according to FER, it can improve user experiences in autonomous cars, marketing strategies, virtual assistants, virtual reality, and video games. Emotion analysis, understanding human behavior in intelligent machines, and social robotics by using face expression recognition play an important role.

Being able to analyze human emotions is useful for human-computer interaction and various fields such as psychology. Psychologists are researching this task to gain information about personality traits, emotional states, and cognitive processes. For instance, they can analyze their patient's mental health conditions, including anxiety, depression, and mental disorders. As a result of these analyses made with the help of

face expression recognition, early diagnosis can be made, and the patient can overcome the disease before even feeling bad.

Face expression recognition has the potential to unlock multiple applications in various fields. The development of precise and robust models for face expression recognition has the potential to make significant improvements in the field of human-computer interaction. Additionally, face expression recognition tasks can facilitate emotion analysis research and improve mental health conditions and support. The immense importance of developing models to recognize facial expressions extends beyond computer vision and has real-world importance for technology, marketing, robotics, psychology, and health care.

1.2 Aim and importance of the study

This thesis aims to examine a comprehensive comparison of the performance of the VGG19, InceptionV3, ResNet152V2, MobileNetV2, and EfficientNetV2B1 CNN architectures on face expression recognition. Through this thesis study, researchers will gain more insight into these models' performance, strengths, weaknesses, and suitability in FER. Face expression recognition systems can be implemented in various applications; thus, different models are used in the comparison.

Non-verbal communication, such as facial expressions, provides important clues about people's mental health or emotions. Recognizing the expressions can access information about one's mental health or emotions. After implementing a stable and accurate face expression recognition system, computers and intelligent machines can improve their experiences, such as user engagement improvement. This improvement directly relates to virtual reality, smart devices, social robotics, and video games.

The comparison of these five different CNN architectures for face expression recognition aims to enhance the development of the field. The findings from this thesis will provide valuable information to researchers and developers to select the most suitable CNN architecture for particular use cases. This thesis aims to gain insight into the most used five CNN architecture's latest versions and conduct this process with challenging tasks such as face expression recognition.

1.3 Original contributions

Numerous original contributions have been conducted to examine face expression recognition tasks and five CNN architectures. Compared to this thesis, the earlier versions of the CNN architectures have been used in face expression recognition studies. However, no studies conducted the latest versions or compared these models in the same conditions.

Multiple research questions are examined with these five CNN architectures. Effects of transfer learning and the importance of pre-training are examined on FER. A comparative analysis was conducted with the Fer2013 dataset and the Faces dataset separately to analyze different kinds of CNN architectures' following characteristics; network depth, convolutional layers, top layer strategies, and how they affect the accuracy of the models.

Examining CNN architectures from common challenges in face expression recognition, such as posed, spontaneous images, and lighting conditions in the images. The analyses performed in this thesis provide detailed information for real-time applications.

1.4 Organization of Thesis

This thesis consists of seven chapters, and each chapter focuses on different components of the face expression recognition task with CNN architectures. The following is the overview of the chapters and their contents;

Chapter 2(p. 6) provides information about neural networks, with detailed information about Feedforward Neural Networks(FFNNs) and Convolutional Neural Networks(CNNs) and transfer learning.

Chapter 3 (p. 18) provides an information about CNN architectures, specially on VGG19, InceptionV3, ResNet152V2, MobileNetV2, and EfficientNetV2B1. These architectures are explained in detail and focused on their characteristics.

Chapter 4 (p. 32) explains the datasets used in FER tasks and the datasets that are used in this thesis; FACES and Fer2013 datasets.

Chapter 5 (p. 36) is a review of the state of the art in this field. It provides information about the studies on FER with CNN architectures.

Chapter 6 (p. 41) explains the methodology and experiments conducted in this thesis. It gives information according to training strategies and results obtained from the experiments. This chapter focuses on four research questions that help to examine the impact of multiple factors in FER tasks on the performance of the model.

Chapter 7 (p. 57) discusses and analysis the thesis findings and its contribution to the field. This chapter gives suggestions for future work.

CHAPTER 2

NEURAL NETWORKS

A neural network(NNs) is a series of machine learning algorithms inspired by human brain operations. That is why they are called neural networks. Neural networks are designed to learn the relationship and patterns in the data without being clearly defined. NNs are able to adapt to changes in the system without being redesigned by the output criteria. NNs consist of layers of interconnected neurons that process and transfer information learned. NNs have a wide range of application fields, including speech recognition, natural language processing, and object detection.

The earliest reference to the development of neural networks is the publication named 'A Logical Calculus of the Ideas Immanent in Nervous Activity' by McCulloch and Pitts from the University of Illinois [24]. The research examined how the human brain understands complex patterns and how it can be simplified to binary logic like true/false or 1/0. This research inspired Rosenblatt to develop the idea of how computers can work with perceptron in 1958[30]. After this research, the concept of backpropagation was found, and research in the field of Neural Networks accelerated. Perceptron is the most straightforward neural network with a single layer, as shown in Figure 2.1

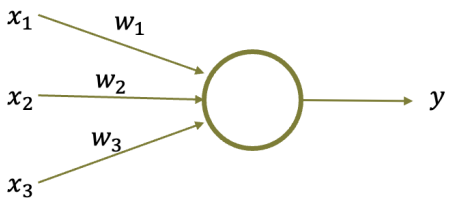


Figure 2.1: Perceptron

After the single-layer perceptron, multi-layered perceptrons are introduced as the input layer, hidden layer, and output layer. The input layer is the first layer in artificial neural networks that accepts input data to the system. The hidden layer is the intermediate layer between the input and output layers. Hidden layers accept weighted inputs from the previous layer and apply the given activation function to them. The output layer is the final layer of the neural network that produces the output predictions of the network. Multi-layer perceptron is shown in Figure 2.2.

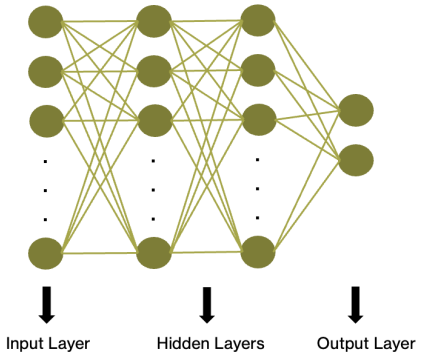


Figure 2.2: Multi Layer Perceptron

Feedforward neural networks 2.1 and convolutional neural networks 2.2 are explained in the next sections.

2.1 Feedforward Neural Networks

The simplest form of NNs is Feedforward Neural Networks(FFNNs). Neural Networks are composed of three layers: input, hidden, and output. The flow of information within these layers is non-linear, starting from the input layer and passing through the hidden layers before reaching the output layer. Feedforward neural networks use weighted connections among neurons, enabling non-linear transformations and learning from data with the help of the optimization process. This section explains fully connected layers, training algorithms, and activation functions and how they help the training process.

The FFNNs are artificial neural networks that process information in one direction, as shown in Figure 2.3. FFNNs do not work in the form of a cycle; it never goes backward like a recurrent neural network. Input data goes through multiple hidden layers and displays predictions in the output layer.

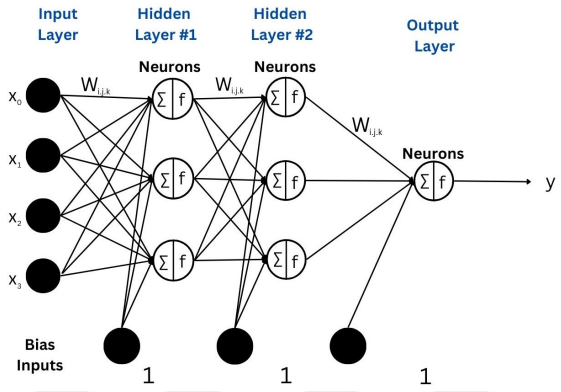


Figure 2.3: Feedforward Neural Networks

FFNNs can be seen as a single layer such as Figure 2.1. Input data enters the model from the input layer and is multiplied by the model weights. After this multiplication, all obtained values from this multiplication sum up to one value.

The fundamental components of FFNNs are the fully connected layers, which are also referred to as dense layers. The layers consist of connected artificial neurons, wherein each neuron in a particular layer is connected to each neuron in the following layer. Fully connected layers are an essential part of the process of face expression recognition, as they are responsible for capturing complex patterns and learning advanced representations from the input data. The configuration of neurons in these layers can be customized to fit the complexity of the given problem.

Activation functions are used in the feedforward neural networks architecture to introduce non-linearity. Introducing the non-linearity helps the neural network observe the model's non-linear relationships between input features and target labels. Multiple activation functions are introduced in the literature, such as sigmoid, Tanh, ReLU, Leaky ReLU, Softmax, Swish, shown in Figure 2.4.

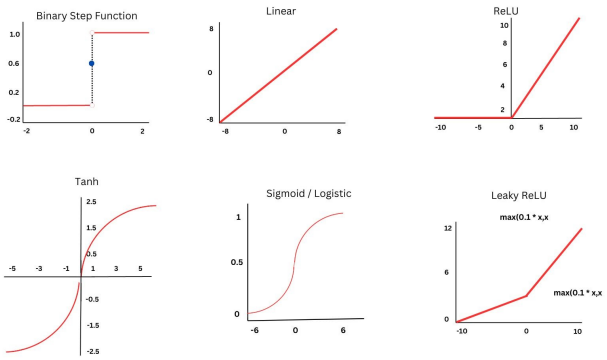


Figure 2.4: Activation Functions

In recent years, Rectified Linear Unit(ReLU) activation function and its versions have become the most popular ones. For instance, x is the input data, w is the weight, and f(x) is the activation function. In a neural network, activation function f(x) is applied with x and w values and transmits to the next layer. The activation function plays a crucial role in the process of recognizing facial expressions. They enable the neural network to understand the representations from input data; thereby, the ability to distinguish between multiple face expressions develops.

The softmax activation function is commonly utilized in the output layer of facial expression recognition models. Figure 2.4 shows the most popular activation functions. Softmax function is that the predicted probabilities add up to one, thereby providing a significant representation of the level of certainty for each class of facial expressions. The implementation of the softmax function is selected over sigmoid in this study due to its advantage in managing multi-class classification tasks. The sigmoid function can be used when there is binary classification.

$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}} \tag{2.1}$$

$$\text{Sigmoid S}(x) = \frac{1}{1 + e^{-x}} \tag{2.2}$$

The final layer of a FFNN shows the final prediction or probability distributions for each class. The final prediction distributions for each class are created by a FFNNs

output layer. In face expression recognition tasks, mainly softmax is used.

Important decisions must be made while training FFNNs, such as the learning rate, optimization algorithm, and performance metrics. The gradient descent optimization step size is determined by the learning rate, which also affects the convergence rate of the learned models.

The weights in the network are updated with the help of commonly used optimization techniques like Stochastic Gradient Descent(SGD) and Adaptive Moment Estimation(Adam). Gradient descent is an iteration algorithm that is used in deep learning to minimize the cost function. It starts from one random point and runs until it reaches the lowest point possible[31].

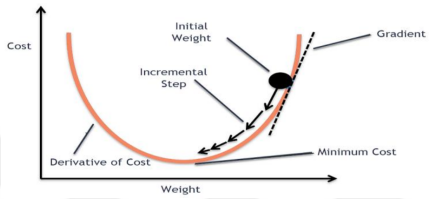


Figure 2.5: Gradient Descent [12]

In the gradient descent algorithm, the whole dataset is loaded at the same time. It causes computational drawbacks. However, stochastic gradient descent(SGD) algorithm derivatives random one data at each step. Thus the computational drawback is avoided.

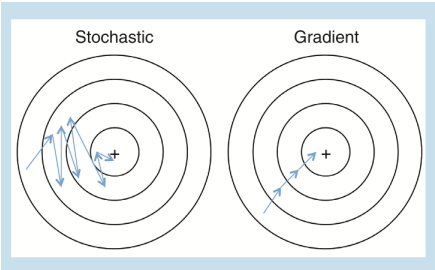


Figure 2.6: Stochastic Gradient Descent and Gradient Descent

Although Adam is recognized for its advantages and faster convergence, there are certain situations where SGD may be preferred over Adam for the purpose of face expression recognition tasks. The SGD optimization algorithm may be considered better than Adam for several reasons.

The SGD algorithm performs better generalization performance, particularly in scenarios where the dataset is limited in size or when working with a limited number of labeled samples. The SGD tends to minimize overfitting by adopting a simpler optimization technique, thus improving the detection of the fundamental patterns in the data. SGD provides improved precision in the learning rate. The learning rate schedule can be modified and fine-tuned while training the model.

When SGD is compared to the Adam algorithm, SGD requires less memory space. The reason is that SGD only needs to store the current gradient, but Adam needs to store the past gradient too. If there is a memory resources problem, using SGD can be beneficial. SGD algorithm allows a more consistent optimization procedure. The update process of SGD is executed through a simple rule that involves adjusting model parameters in response to the gradient of the current mini-batch. It prevents fluctuations or jumps in the cost function.

It is important to say that the success of SGD and Adam changes according to the dataset structure, network architecture, and other elements that are used to train the model. In most of the research, it is recommended to test several optimization techniques and hyperparameters before determining which ones work best with particular facial expression recognition tasks.

Evaluation of the classification performance of FFNNs for facial expression recognition are performance metrics, accuracy, and confusion matrix. These offer essential information about how successfully the model can categorize multiple facial expressions. Accuracy is one of the most crucial performance indicators for classification tasks. Validation accuracy is the percentage of correctly classified data in the validation set. It provides overall knowledge about how successfully the model classified unknown data and accurately recognized face expressions. Accuracy results give an important outcome about the performance of the model. It is important for the model's ability to generalize and predict real-world images. A high validation accuracy means that the model becomes familiar with patterns and characteristics associated with facial expressions. Other than evaluation of validation accuracy, determining from the confusion matrix is also a common technique for evaluating the performance of the model.

Validation accuracy performs as a biased measure in this study. The validation accuracy results that are shown in the tables and also used for comparison are selected

from the highest validation accuracy result from the epochs during training, which makes the validation accuracy biased.

Training accuracy shows the evaluation of the model predictions on the training data. It illustrates how much percentage of the training dataset and true labels predicted succesfully. While high validation accuracy indicates that the model makes predictions correctly, high training accuracy does not always mean good results. Training accuracy is monitored throughout the model's training to evaluate how well the model is converging.

Converging means performance improvement in the training data. Training accuracy should increase while the model is still learning the patterns from training data. However, it is crucial to understand that training accuracy does not always gives high results on unseen data. Very high training accuracy sometimes means that the model is not learning but memorizing the images.

When the model starts to memorize the training images, it is called overfitting. Overfitting occurs when the model only becomes familiar with training data and does not perform well on validation data. That is why datasets are divided into training, validation, and test sets, to overcome the constraints of training accuracy.

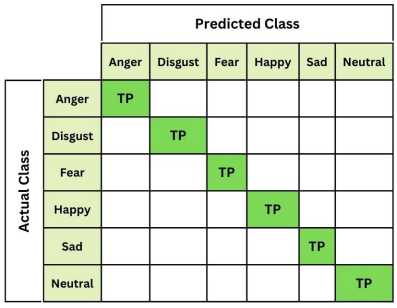


Figure 2.7: Confusion Matrix

Confusion matrix offers comprehensive and detailed information about predictions of the model for each class. The presented Figure 2.7 displays accurately classified classes, referred to as true positives, as well as the number of incorrectly classified classes in the white boxes.

By looking at the model's confusion matrix, it is possible to observe which classes the model is misclassified. The model's difficulty in discriminating can be reduced

with the help of a confusion matrix, and its performance can be improved. The previously mentioned evaluation has the potential to offer significant insights that can help optimization of subsequent models.
The confusion matrix offers a deeper understanding of the model's performance for individual classes in the FFNNs. In conclusion, these metrics make evaluating FFNN's ability to classify facial expressions in recognition tasks easier.
Feed Forward Neural Networks that include a fully connected layer, required activation function, a softmax output layer, and appropriate training hyperparameters have potential in the task of face expression recognition.
To summarise, after learning the importance of mentioned components and their importance to the recognition of facial expressions, researchers and developers can develop functional architectures for FFNNs, and analyze the model performance with the metrics such as accuracy and confusion matrix.

2.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a category of neural network structure that is designed to process and analyze visual data, such as images. CNNs consist of multiple layers, such as a convolutional layer, a pooling layer, batch normalization, dropout, and a fully connected layer. These layers work in a combined way to identify the particular features in the input images and produce predictions.
The reason behind the use of CNNs, is their capability to capture spatial hierarchies and local features on the image. Convolutional layers are the main part of the CNNs. Convolutional layers apply filters, alternatively called kernels, that move goes on the input image and executes element-wise multiplication and summation to generate feature maps. CNN identifies local patterns and characteristics on the image, regardless of their spatial position, through the use of filters. The sharing of parameters allows productive learning and decreases the number of parameters in the neural network. In this thesis, RGB images are used, which have three channels. The number of channels in the input defines the depth of the CNN. The depth changes through the network with the previous layer in the CNN. In Figure 2.8, the left part of the figure represents kernels applied to the image, and the right side is the 2D activation

map output. Each kernel goes through the image, applies element-wise multiplication and summation, and then stores the activation map. The network learns when kernels see a specific feature in the spatial domain. The deeper the network goes, the more filters can activate only in the high-level features in facial images, such as eyebrows. Pooling layers perform a down-sampling operation on the feature maps,

Figure 2.8: Convolutional Layer

thereby decreasing spatial dimensions while restoring the most significant features. The down-sampling process helps the maintenance of relevant data while eliminating less informative data. This enables the network to concentrate on important features and reduces computational drawbacks.

In specific computer vision tasks, such as facial expression recognition, a Global Average Pooling (GAP) provides significant advantages. The GAP layer is a pooling layer that computes the average output of each feature map present in the layer that leads to it. When compared with other pooling layers, the GAP layer approach represents a combination of spatial data across the entire feature map and rendered spatial data. The use of the GAP layer in tasks such as facial expression recognition can be related to several factors, as explained. Applying global average pooling in deep learning models reduces overfitting, an important problem. By reducing the overall number of parameters in the model by converting a 2D feature map into a singular representation. GAP demonstrates robustness to spatial transformation with the ability to collect the entire spatial information of a feature map. This characteristic of GAP is especially important in the task of face expression recognition, where input expressions can be presented in different spatial locations across different images. The GAP technique establishes a direct connection between the feature maps and the output classes, thus enabling the neural network to consider the complete image while making a decision instead of prioritizing the most activated parts.

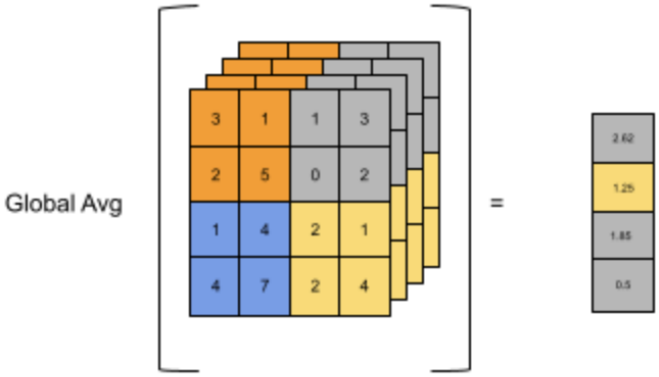


Figure 2.9: Global Average Pooling[22]

Based on the obtained results, global average pooling can potentially improve the model's accuracy to a higher level than other learning layers. The GAP layer helps improvement of the validation accuracy, especially in computer vision tasks such as face expression recognition. This layer is often added between sequential convolutional layers in CNNs. The mismatch in the network is controlled with the help of GAP.

Usually, convolutional and pooling layers are followed by non-linear activation functions, such as Rectified Linear Units (ReLU). Integrating activation functions in convolutional neural networks (CNNs) introduces non-linear characteristics to the network, allowing the model to learn complex patterns between features and improving the model's representation capacity.

In the end, the results obtained from convolutional and pooling layers flattened and transmitted to fully connected layers. Each neuron in these classes is connected to each neuron in the following layer, thus allowing extensive interactions and collecting high-level representations. This is why the fully connected layer is the last and the most important layer in the network. In this layer, the neuron performs a linear transformation to the input vector by using a weight matrix. A non-linear activation function f(x) is used to apply a non-linear transformation on the product.

The fully connected layers are responsible for capturing the global patterns that exist among the features extracted by previous layers. These relationships are then used to make predictions based on the representations that have been created.

2.3 Transfer Learning

Transfer learning is a deep learning approach that stores the knowledge gained while learning the problem and uses that knowledge when faced with a similar problem. This approach implements a pre-trained model, which is trained on a large dataset such as ImageNet. It helps its performance when there is a similar task that involves a smaller dataset. Transfer learning is an approach that uses existing knowledge and applies learned representations of pre-trained models to speed up and improve performance in the training process.

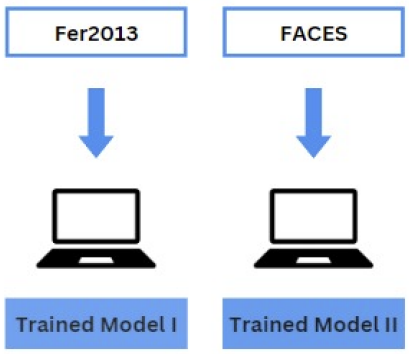


Figure 2.10: Traditional Learning Approach

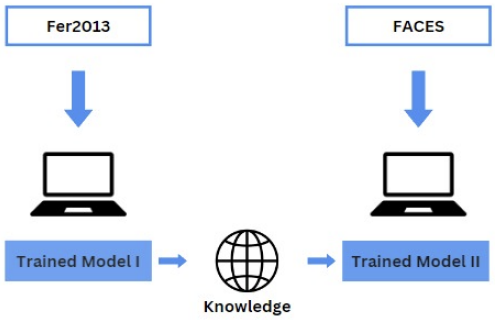


Figure 2.11: Transfer Learning Approach

Transfer learning uses a pre-trained model to extract features, whereby the model captures general patterns and high-level representations from the original set of data it was trained on. The learned features can potentially be beneficial when dealing with similar tasks. The approach accelerates the learning process of new tasks without

considering the new dataset is whether small or not.

Transfer learning consists of two steps; pre-training and fine-tuning. In the pre-training step, a neural network model is trained with a large dataset, such as ImageNet, that contains millions of labeled images from various classes.

This training process allows the model to learn fundamental features and higher-level representations that can be applied to a wide range of visual tasks.

In the fine-tuning step, the pre-trained model is modified for the intended task by adjusting the parameters that are suitable for a new task.

The process includes the adjustment of the existing model's learning layers to customize the model's output for desired classes. The weights of pre-trained models are either frozen or adjusted with a lower learning rate. However, the weights of the newly added layers are updated during fine-tuning.

The use of transfer learning offers various advantages. For instance, using features from a pre-trained model results in remarkable conservation of computational resources and training duration. The reason for this, the model uses the knowledge gained from the previous training and does not need training from scratch.

Transfer learning solves the problem of inadequate labeled data by using knowledge gained from a larger dataset. The model's performance can be improved using transfer learning, especially in tasks where labeled data is insufficient or the dataset is extensive, and training from scratch is time-consuming. Transfer learning enables developers to develop models by using knowledge from the pre-trained models that experts in the respective field develop. To summarise, transfer learning is an approach that facilitates the transfer of learned knowledge from pre-trained models to novel tasks. Transfer learning improves models' effectiveness, adaptability, and efficacy for their tasks. It helps to make valuable strategies in various computer vision tasks such as face expression recognition.

This thesis explores the CNN architecture's performances by using transfer learning in the task of facial expression recognition. In the following sections, CNN architectures and how transfer learning is applied to models are explained in detail.

CHAPTER 3

CNN ARCHITECTURES

This chapter provides information about Convolutional Neural Networks (CNNs) and their applications across various domains. This section will comprehensively cover CNN architectures, including their fundamental principles, operational mechanisms, and their particular applicability to the face expression recognition task.

The training of CNN architectures has become a fundamental operation to analyze image data, especially in the field of computer vision tasks. The previously mentioned VGG19, ResNet152V2, InceptionV3, MobileNetV2, and EfficientNetV2B1 CNN architectures are developed to extract complex features from given image data, enabling intelligent machines to understand complex patterns in the learning process. Convolutional Neural Networks are composed of convolutional, pooling, and fully connected layers. These networks show a remarkable ability to learn hierarchical representations, allowing them to accurately recognize important features in image data.

This thesis focuses on five different kinds of CNN architectures, called VGG19, InceptionV3, ResNet152V2, MobileNetV2, and, EfficientNetV2B1. Each of these models has distinctive characteristics that make them highly suitable for face expression recognition task. Each of these models has its own characteristics that make them suitable for different conditions in the training process.

The choice of the CNN architectures that can be implemented in face expression recognition depends on several factors, such as the type of the dataset, the availability of computational resources, and the particular requirement of the task. When training this CNN architecture, an optimal experimental setup is conducted that works for each of the CNN architectures and gives high results. Each CNN architecture is explained in detail in the following sections; VGG19 in section 3.1, InceptionV3 in section 3.2,

ResNet1512V2 in section 3.3, MobileNetV2 in section 3.4, and EfficientNetV2B1 in section 3.5.

CNN	Size(MB)	Number of Layers	Number of Parameters
VGG19	549	19	143.7M
InceptionV3	92	189	23.9M
ResNet152V2	232	307	60.4M
MobileNetV2	14	105	3.5M
EfficientNetV2B1	34	337	8.9M

Table 3.1: CNN Architecture Information

3.1 Visual Geometry Group 19

Simonyan and Zisserman developed VGG19 CNN architecture at the Visual Geometry Group (VGG) located at the University of Oxford[35]. In 2014, Karen Simonyan and Andrew Zisserman published their article about VGG architecture. The article entitled "Very Deep Convolutional Networks for Large-Scale Image Recognition"[35]. The VGG architecture model has gained immense attention and has been extensively used in various computer vision tasks, such as image classification, object detection, and image segmentation. The model showed exceptional performance during the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2014, achieving the best performance in the localization task and the second best in the image classification task.

The fundamental architectural design underlies VGG19, including the consecutive order of convolutional layers that applies small 3x3 filters, followed by max-pooling layers, which are used to reduce spatial dimensions. This consecutive order in VGG19 CNN architecture allows the advancement of deeper neural networks with extended receptive fields.

As a result of the VGG19 CNN architecture design, complex patterns, and hierarchical representations shown in the input image can be captured successfully. The latest version of VGG architecture is VGG19 which is used in this thesis. The VGG19 ar-

chitecture consists of a total of 19 layers, which include 16 convolutional layers and three fully connected layers. Architecture is shown in Figure 3.1.

Figure 3.1: Architecture of VGG19 model

The VGG19 model is pre-trained with an ImageNet dataset that consists of 1000 classes. Each of the convolutional layers applies 3x3 filters. Five groups of convolutional layer sets are applied to the input images between the max pooling layer. After the max pooling layer is applied, a reduction of spatial resolution by a factor of 2 can be observed. VGG19 architecture accepts 224x224 input images.
However, the model input shape can be modified to desired input shape. VGG architectures come with fully connected layers before modifying them. The fully connected layers work as a classifier by displaying the final predictions. Therefore, fully connected layers are located at the end of the network.
VGG19 is known for its uniform and simple architectural design by only using the same filter and layers consecutively throughout the whole network. However, number of parameters is the highest among the models used in this thesis. Under this simple structure, there are hidden extra memory usage and computational intricacy problems laying. A high number of parameters makes it resource-intensive when compared to all the other CNN architectures. The reason why VGG19 has a high number of parameters even though it has 19 layers is that in each convolutional layer, it has 3x3 filters with a stride of 1 and padding of 1. The number of filters increases when models go deeper. The number of parameters can be seen in Figure 3.2.

VGG19 architecture showed its success in multiple visual classification tasks, especially in cases where a large number of labeled training data is used and has a diverse number of examples in its classes. The deep and hierarchical structure of the system allows the model to gather comprehensive features from images, both low-level meanings and high-level meanings.
As previously mentioned, VGG19 architecture has been popularly implemented and demonstrated excellent performance in various computer vision tasks[23], [1], [13],

Layer name	#Filters	#Parameters	#Activations
input			150K
conv1_1	64	1.7K	3.2M
conv1_2	64	36K	3.2M
max pooling			802K
conv2_1	128	73K	1.6M
conv2_2	128	147K	1.6M
max pooling			401K
conv3_1	256	300K	802K
conv3_2	256	600K	802K
conv3_3	256	600K	802K
conv3_4	256	600K	802K
max pooling			200K
conv4_1	512	1.1M	401K
conv4_2	512	2.3M	401K
conv4_3	512	2.3M	401K
conv4_4	512	2.3M	401K
max pooling			100K
conv5_1	512	2.3M	100K
conv5_2	512	2.3M	100K
conv5_3	512	2.3M	100K
conv5_4	512	2.3M	100K
max pooling			25K
fc6		103M	4K
fc7		17M	4K
output		4M	1K

Figure 3.2: Detailed Architecture of VGG19 Model[10]

[28], [16], such as facial expression recognition (FER). Deep CNN architectures such as VGG19 have popularity because of their capacity to capture complex patterns. VGG19 architecture showed successful results while identifying the facial expressions from Fer2013 and FACES datasets.

The reasons why VGG architecture showed such high accuracy are explained as follows; ability to capture both low-level facial features like edges and texture and high-level semantic information like facial landmarks and expressions[35],[20]. Facial landmarks represent key features of the face, such as eyebrows, nose, jawline, mouth, and eyes. Facial landmarks play an important role in multiple tasks such as calculation of facial symmetry, estimation of pose, and recognition of face expressions[20].

The dataset used to train the model also has an immense impact on the model performance. One of the datasets is Fer2013. The Fer2013 dataset is used in the training part of the thesis. The Fer2013 dataset is used to evaluate the performance of the VGG architecture for the face expression recognition task.
Several studies showed that VGG19 has the capability to achieve the highest accuracy with the FER2013 dataset. Some researchers achieved to get above %70 accuracy and have exceeded the current best performance with VGG19 architecture in the literature. Section 5 provides a more detailed explanation of the state of the art belonging to face expression recognition and 5 CNN architectures used in this thesis.

It is notable that even though VGG19 demonstrated outstanding accuracy in face expression recognition tasks, there are other architectures used in this field. Multiple CNN architectures, including ResNet, Inception, MobileNet, and EfficientNet, have been used and have shown competitive performance in face expression recognition tasks.

3.2 InceptionV3

In 2015, Google researchers introduced a convolutional neural network(CNN) architecture named InceptionV1/GoogleNet that consists of 22 layers [37]. They published the first publication that uses the term width/wider in convolutional neural networks. The module called inception is the module that makes models wider.
Inception architecture has its own architectural design. The Inception modules consist of convolutional layers with several filter sizes of 1x1, 3x3, and 5x5 that perform the multiple number of pooling in parallel. This inception module is called Naive Inception Module, which is shown in Figure 3.3. The dimensionality reduction version of the inception module is developed due to the fact that too large dimensionality size after the naive version. After adding the 1x1 convolutional layers before to the naive version layer fixed that problem by decreasing the dimensionality. A new version of the inception module is shown in Figure 3.4.
The detailed architecture of the first version of Inception is shown in Figure 3.5. The researchers who developed Inception architecture later suggested that it could be developed into a more practical version.

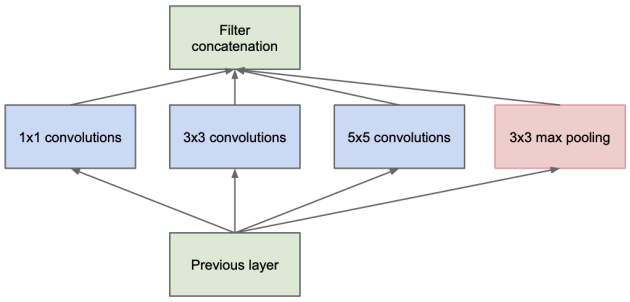


Figure 3.3: Inception module, naive version [37]

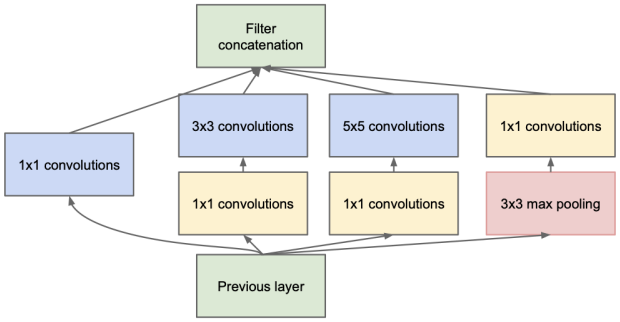


Figure 3.4: Inception module, dimentionality reduction [37]

type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

Figure 3.5: Architecture of InceptionV1/GoogleNet [37]

In the InceptionV1 architecture, 1x1 convolutional dimensionality reduction was applied before 3x3 convolutional layers. The training process can be applied faster by

multiplying the convolution process with a broader filter. Convolutional operations with large dimensions, such as the 5x5 convolutional layer in Figure 3.4, cause a computational burden.

The InceptionV3 CNN architecture is developed to solve these problems in the earlier version [38]. It is indicated that reaching the desired dimension with two smaller convolutional layers is better than a bigger one. Computation is saved %33 in this version of the Inception module, which is shown in Figure 3.6.

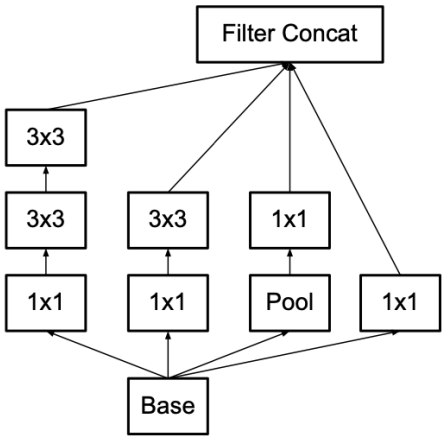


Figure 3.6: Inception modules where each 5x5 conv is changed with two 3x3 conv [37]

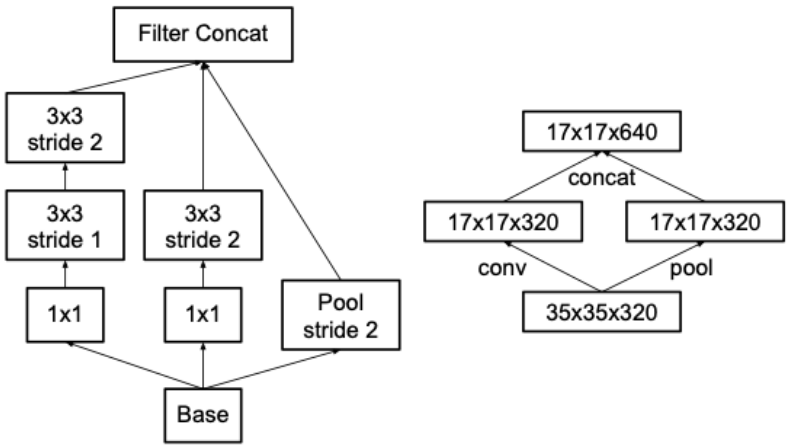


Figure 3.7: Inception module used in InceptionV3 [37]

Finally, the latest version of the Inception module has been developed. According to the researchers, when the (nxn) convolutional layers are divided into two layers (1xn) and (nx1), the computational load decreases as n. This approach showed incredible results—the latest version of the Inception module is shown in Figure 3.7.

These updates are applied to improve both the accuracy and efficiency of the classification of images. In InceptionV3, parallel convolution operations and pooling operations are applied with the stride of 2. The computational load is reduced with the help of this process. The Inception architecture is able to be more effective than other CNN architectures, such as VGGNet.

To summarise, the latest version of Inception is a popular model that is used in various computer vision tasks such as image classification and object detection.
The architecture of the InceptionnV3 can be examined by looking at two different pattern recognition. Inception modules use various filter sizes. For example, 3x3 convolutional filters are used to capture local features in the input images. For instance, 5x5 convolutional filters have the ability to detect edges, corners, and small-scaled features in given image data. Combining these multiple filter sizes in a single layer leads to the recognition of a wide range of local features.
Another pattern recognition is called global pattern recognition. The neural networks can capture global patterns with the help of 1x1 convolutional filters in Inception modules. As mentioned before, a convolutional filter by the size of 1x1 operates as channel-wise dimensionality reduction. It efficiently decreases the number of input channels. For example, a convolutional filter with the size of 1x1 can capture the overall structure of the entire image.
In total, InceptionV3 architecture has 189 layers and 23.9 million parameters. When the InceptionV3 model is pre-trained with ImageNet weights, the model can detect up to 1000 classes. The input size of the model is 299x299 pixels.
Considering all of these characteristics of the Inceptionv3, such as the ability to capture both low-level and high-level features in the input image, shows that this architecture can perform high accuracy with complex training and test images.

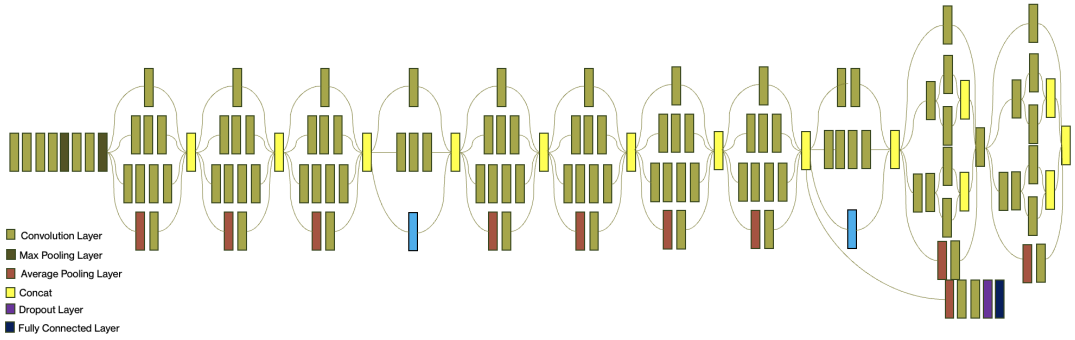


Figure 3.8: InceptionV3 Architecture

3.3 ResNet152V2

ResNet152v2 convolutional neural network architecture design comes from Residual Networks(ResNet). Residual Network is a deep learning architecture specially developed for computer vision tasks.

The architecture is composed of convolutional layers, batch normalization layers, and fully connected layers. The architecture of the ResNet is developed to aid thousands of convolutional layers.

Previously developed architectures were incapable of handling thousands of layers. After adding a high number of layers, the model resulted in a 'vanishing gradient' problem. The vanishing gradient problem was a restriction for performance.

The training of neural networks includes a backpropagation process that depends on gradient descent. The backpropagation process is directly related to decreasing the loss function and detecting the weights that minimize gradient descent.

Such a large number of layers causes repeated multiplications. Repeated multiplications in architecture eventually result in the disappearance of the gradient descent. The performance of architecture becomes saturated or degrades with every extra layer. ResNet found a novel solution to the vanishing gradient problem, skipping the connections. The architecture uses the technique of stacking multiple identity mappings, composed of convolutional layers that previously did nothing, and afterward skip over these layers while reusing the activations of the previous layers.

All layers belonging to the architecture are extended, and the rest of the architecture, which is called residual parts, are permitted to explore more feature space than before.

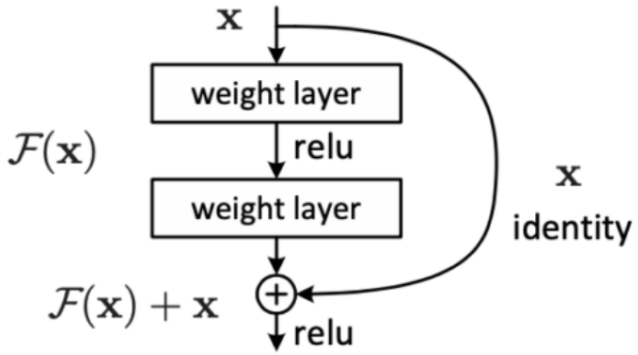


Figure 3.9: Residual Blocks[14]

As shown in Figure 3.9, a residual operation can be expressed as output = F(x) + x. In this formula, x represents the input to the block that is also expressed as the last layer coming from the previous layer. F(x) represents the convolutional blocks. Residual block helps to reduce the gradient flow during backpropagation. This way, architecture can go up to hundreds or thousands of layers without gradient vanishing problems.

Pre-activation variant of residual blocks is applied, which can be explained as the application of activation functions and batch normalization before the convolutional layers. Pre-activation is one of the techniques that helps models to minimize the vanishing gradient descent.

In face expression recognition, there are unique characteristics that the model should focus on, such as eyes, mouth, and facial curves. ResNet architecture can capture both local and global features from faces.

In this thesis, the latest version of Residual Networks is used as one of the convolutional neural networks. ResNet152V2 has 307 layers, and the number of parameters is 60.4 million. The structure of ResNet152v2 showed outstanding performance in extracting informative features from given image data.

3.4 MobileNetV2

MobileNetV2 is an updated version of MobileNet and a convolutional neural network architecture. This model is specially designed for mobile and embedded systems such as raspberry pi and Arduino.

This model involves two new concepts: inverted residuals and linear bottlenecks. MobileNetV2 model has a simple architecture that uses residual connections around bottleneck layers. The neural network starts with 32 filter sizes and goes through 19 residual bottleneck layers. The last layers are a 1x1 convolutional layer, global average pooling, and a softmax layer classification layer.

Residual blocks consist of three layers; a bottleneck expansion layer, a depthwise convolution layer, and a projection layer which is known as a fully connected layer. Depthwise separable convolutions work by replacing a fully convolutional operator with a factorized variant that divides convolution into two layers.

The initial layer is a depthwise convolution that executes lightweight filtering via a single convolutional filter for each input channel.

The second layer consists of a convolutional layer with a size of 1x1, referred to as pointwise convolution. The pointwise convolution layer generates new features by performing linear combinations of input channels.

Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	

Figure 3.10: MobileNetV2 Architecture[33]

MobileNetV2 architecture consists of two key points; inverted residuals and linear bottlenecks. The meaning of 'inverted' refers to the characteristic of the residual block. Figure 3.11 shows differences between the residual block and inverted residual blocks. Diagonally hatched layers do not use non-linearities. The thickness of the blocks is used to indicate the relative numbers of channels. As shown in the figure, classic residual blocks connect a large number of channels, and inverted residual blocks connect only bottlenecks.

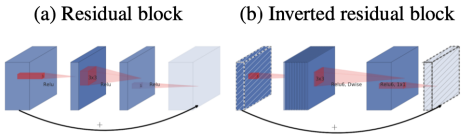


Figure 3.11: Residual Blocks and Inverted Residual Blocks[33]

3.5 EfficientNetV2B1

EfficientNetV2B1 is one of the EfficientNetV2 models. EfficientNetV2 models are specially developed for image classification problems. EfficientNetV2 models were developed by Google researchers Mingxing Tan and Quoc V. Le in 2019, and the models are advanced versions of the EfficientNetV1 models[39].

The novel approach behind the latest version of EfficientNet is balancing the model size, accuracy, and speed. This balance is accomplished by using scalable architecture and new training methods. Tan and V. Le introduced a newly developed technique that helps the model to enhance its performance, such as scaling, Fused-MBConv, and progressive learning. Compound scaling is a technique that scales a model's depth, width, and resolution. In this study, depth represents the number of layers, and width is the number of channels in the model. In the EfficientNetV1 model, researchers proposed a new compound scaling method that is shown in equation 3.1. The compound coefficient is used to scale the network depth, width, and resolution.

$$
\begin{aligned}
&\text{depth: } d = \alpha^{\phi}, \\
&\text{width: } w = \beta^{\phi}, \\
&\text{resolution: } r = \gamma^{\phi} \\
&\alpha \cdot \beta^2 \cdot \gamma^2 \approx 2 \\
&\alpha \geq 1, \beta \geq 1, \gamma \geq 1
\end{aligned}
\tag{3.1}
$$

The values of α, β, and γ can be determined through a small grid search. The compound coefficient is represented by φ, and the user should select it. This coefficient is used to control how many resources are allocated for the specific model scaling. After deciding the coefficient, the model's parameters, number of layers, channels, and resolution can be seen.

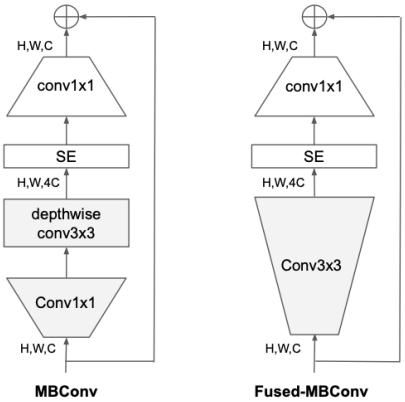


Figure 3.12: MBConv and Fused-MBConv[39]

Another technique updated with the new model is neural architecture search(NAS). As shown in Figure 3.12, the previous version had a mobile inverted bottleneck structure, an extended version of MBConv. MBConv had a convolution layer of size 1x1, squeeze and excite block, and depthwise convolution layer with size of 1x1. The previous version of this model uses an inverted residual bottleneck, as already explained in Chapter 4.3.

However, training with large MBConv structure images is slow, which was a significant drawback for this structure. The only solution for this problem was decreasing the batch size to accommodate the large images, but this is not the optimal solution. Another drawback is that the model's training takes too much time because of the

depthwise operations. In addition to these, in compound scaling, changing the value of the coefficient causes all of the values to change, and this leads to a more uncontrolled training process.

EfficientNetV2 proposed a new approach to training NAS. The model needs the best accuracy(A), less training step time(S), and the most important one is that the model should perform with fewer parameters(P).

$$F = A \cdot S^w \cdot P^v \tag{3.2}$$

w and v are the constants that can be determined by the user. In the EfficientNetV2, the depthwise convolution layer and 1x1 convolution layers are combined, and it is called Fused - MBConv. Fused - MBConv can be applied in the early, middle, or all of the model. The user should determine the stage of the Fused-MBConv. For example, the Fused - MBConv layer can be applied in the early stage of the model and gives better training than before but worse training results than fully applied Fused-MBConv. It depends on the user and the results of the experiments.

EfficientNetV2B1 has the best scalable design and robustness to variations. After careful training experiments, EfficientNetV2B1 is the most suitable version of EfficientNets for face expression recognition with FER2013 and FACES datasets.

CHAPTER 4

DATASETS

In this chapter, face expression recognition datasets will be explained. Face expression recognition systems can be implemented in two distinct categories. One of these categories is working with dynamic image sequences, where a neural network processes consecutive frames to gather information over time.

However, this thesis adopts the second category, which utilizes static images for the models. In this category, the neural network analyzes each image without considering information across multiple frames.

In this field, several datasets have been widely used to evaluate convolutional neural networks. Each of these datasets has its own characteristic. This diversity can be due to the structure of the image, the type of image, and even the way it was taken. The datasets used in this study and other frequently used datasets by researchers are shown in Table 4.1.

In the literature, there are lots of datasets that only contain face expressions. Some of these datasets consist of 6 basic facial expressions, while others consist of 7 facial expressions with the addition of surprise expressions. The number of subjects in datasets such as CIFE, Fer2013, and EmotioNet, created by collecting images publicly available on the internet, is unknown. The number of samples of the datasets, which means how many images they contain, varies in a wide range.

When the average number of samples is considered, Fer2013 is much ahead of the other datasets in terms of sample size. Some datasets in the literature were posed in labs, while others were taken in spontaneous poses. All of these datasets provided researchers that work in the field of face expression recognition with labeled face images. This allowed researchers to train and test convolutional neural network mod-

els efficiently. The upcoming sub-chapters will provide comprehensive information about the Fer2013 and FACES datasets used in this thesis.

Database	Number of Subjects	Number of Samples	Environment	Elicitation Method	Number of Expressions
Fer2013	N/A	35,887	Web	Posed+Spon.	7
FACES	171	2,052	Lab	Posed	6
JAFFE	10	213	Lab	Posed	7
CK+	123	593	Lab	Posed+Spon.	8
Lifespan	576	1,046	Lab	Posed	3
CIFE	N/A	14,756	Web	Posed+Spon.	7
KDEF	70	4,900	Lab	Posed	7
EmotioNet	N/A	1,000,000	Web	Posed+Spon.	23

Table 4.1: Information about face expression datasets

4.1 Fer2013 Dataset

The Fer2013 dataset is a popular and commonly used dataset in the field of face expression recognition. It was first created and introduced in 2013 in Challenges in Representation Learning[11].

The Fer2013 dataset consists of 35,887 grayscale images, each with a size of 48x48 pixels. The dataset is publicly accessible to everyone. The images were collected from various sources on the internet and as a result, the exact number of subjects remains unknown. These images contain spontaneous poses from the subjects. The Fer2013 dataset contains 28,709 training images, 3,589 validation images, and 3,589 test images with six basic expression labels; anger, fear, disgust, happiness, sadness, neutral, and additionally surprised. The dataset provides an unbalanced number of samples for each expression. This unbalanced structure provides valuable resources for evaluating and comparing deep learning models in face expression recognition.

Figure 4.1: Example Images From Fer2013 Dataset

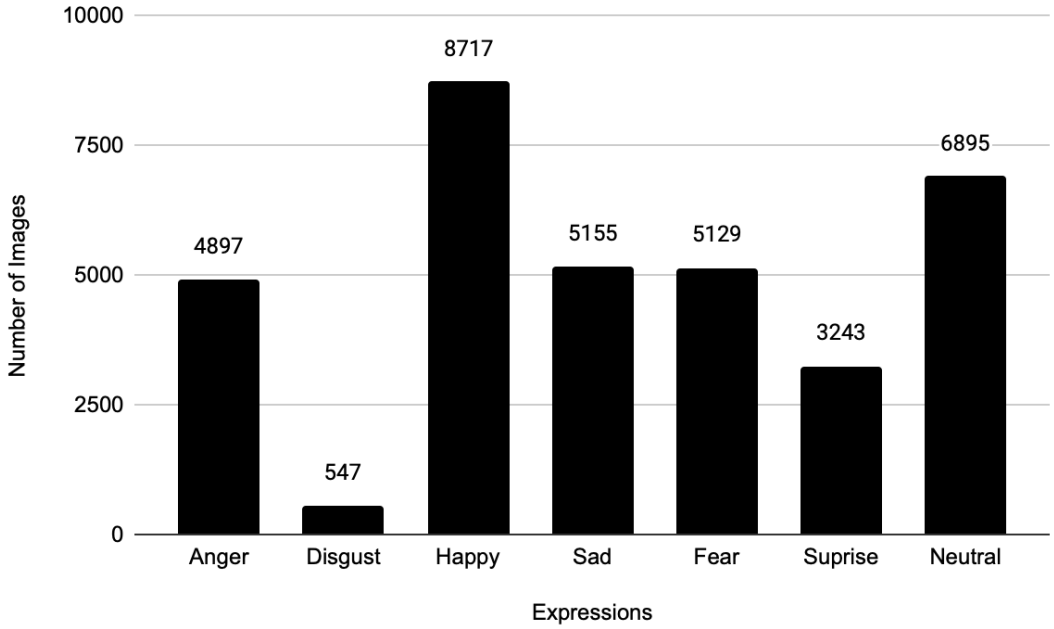


Figure 4.2: Fer2013 Expression Distribution

4.2 FACES Dataset

The FACES dataset is not a publicly available dataset and was introduced by Max-Planck Institute for Human Development in 2010[8]. It consists of a total of 171 naturalistic faces belonging to individuals of varying ages and genders, including young, middle-aged, and older women and men.

The images in the dataset were captured in a lab environment, and each of the images is posed. The subjects stand in the same environment with consistent backgrounds, poses, and lighting on their faces. Each image has a resolution of 2835 x 3543 pixels. Additionally, each subject posed twice for each expression, resulting in 342 images per expression for a given subject.

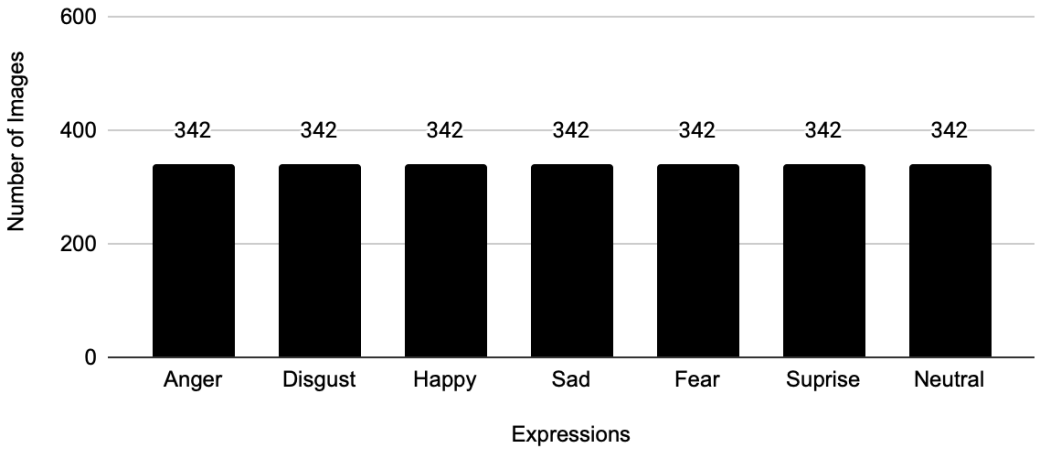


Figure 4.3: FACES Expression Distribution

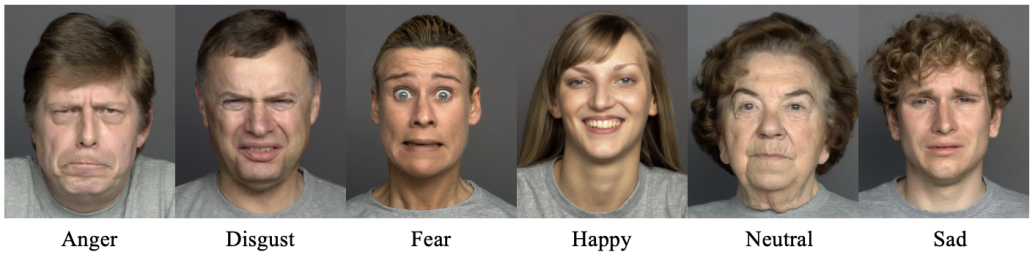


Figure 4.4: Example Images From FACES Dataset

CHAPTER 5

STATE OF THE ART

The field of face expression recognition(FER) has attracted attention from researchers in computer vision and artificial intelligence. Over the past few decades, this field has gained significant attention. Face expression recognition studies have garnered more attention and interest compared to other academic research fields.
FER finds applications in diverse areas such as human-computer interaction, healthcare diagnostics in psychology, video game rating, driving assistance, and interviews. The capability to determine a person's emotional state through facial expressions without verbal communication holds immense potential for research and development in these fields.

5.1 Background

Researchers have made remarkable progress in the field of face expression recognition and showed that there is a promising path to follow for future research. Various neural network architectures have been used in this study, including VGG16, InceptionV3, MobileNet, and EfficientNet. Each of these models demonstrated unique advantages and contributed novel perspectives to the study.
Face expression recognition is not only a subject of interest in computer science but also in psychology. The complexity of this task arises from the fact that individuals can express multiple expressions simultaneously or at the same time[7]. This is one of the main factors that make this task challenging.
Face expression recognition studies started with studies of Paul Ekman, who introduced facial expressions are universal and it does not depend on cultural and social

differences[9]. Face expression recognition studies inspired by the theories of Ekman. After Ekman's theory was introduced, the discovery of techniques such as face extraction techniques, traditional machine learning, and image processing techniques gained momentum.

In the past, face expression recognition (FER) studies involved manual feature extraction from facial images, including geometric features and landmarks, before the advent of deep learning techniques. After this process is finished manually, researchers used these features to train Support Vector Machines(SVMs), Decision Trees, and Bayes classifiers[2]. However, these classical methods have limitations because of the fact that it affects quality and resolution.

In the current decade, classical approaches have been replaced mainly by deep learning methods. Deep learning models automatically learn relevant features from raw pixels, thereby eliminating the need for manual feature extraction. Various state-of-the-art deep learning models have been developed for FER, such as VGGFace and ResMaskNet. Each model adopts unique approaches, such as residual connections, bottleneck layers, and attention mechanisms, to achieve higher accuracy in face expression recognition[29]. These models are often trained, fine-tuned, or tested using face expression recognition datasets like Fer2013, CIFE, and CK+. Some of these studies have shown very satisfying results with the help of these datasets.

Dataset	Method	CNN Architecture	Transfer Learning	Pre-training Dataset	Pre-processing	Data Group	Additional Classifier	Performance(%)
Fer2013	Pramerdorfer et al. 16[27]	VGG	-	-	-	Train, Test, Validation	-	%72.70
	Pramerdorfer et al. 16	ResNet	-	-	-	Train, Test, Validation	-	%72.40
	Pramerdorfer et al. 16	Inception	-	-	-	Train, Test, Validation	-	%71.60
	Khanzada et al. 20[18]	VGG16	+	VGGFace	+	Train, Test, Validation	-	%70.20
	Khanzada et al. 20	ResNet50	+	VGGFace	+	Train, Test, Validation	-	%73.20
	Khanzada et al. 20	Custom-model	-	-	+	Train, Test, Validation	-	%66.30
	Kusuma et al. 20 [19]	VGG16	+	ImageNet	+	Train, Test, Validation	-	%69.40
	Caroppo et al. 19[17]	VGG16	+	ImageNet	-	Train, Test, Validation	RF	%71.50
	Khaireddin et al. 21[17]	VGG	-	-	-	Train, Test, Validation	-	%73.28
	Hua et al. 19[15]	VGG19	+	ImageNet	+	Train, Test, Validation	-	%62.31
FACES	Sajjanhar et al. 18 [32]	InceptionV3	+	ImageNet	-	Random Split	-	%82.19
	Sajjanhar et al. 18	VGG19	+	ImageNet	+	Random Split	-	%97.16
	Sajjanhar et al. 18	VGGFACE	+	ImageNet	-	Random Split	-	%95.06
	Caroppo et al. 19[4]	VGG16	+	ImageNet	-	Random Split	RF	%97.21

Table 5.1: State-of-the-art studies in FER

5.2 CNN models trained with Fer2013 dataset

As stated in previous chapters, the Fer2013 dataset is a widely used dataset in FER studies due to its unbalanced distribution, large number of images, and comparability with existing literature[11]. The low resolution of the Fer2013 dataset, which contains a different number of images in each class, and the high resolution of the FACES dataset, which contains the same number of images in each class, provided an opportunity to examine the balanced/unbalanced dataset structures on the models. It is observed that datasets created in controlled lab environments, such as Faces, often achieve accuracy around 80%, whereas fixed web datasets, like Fer2013, typically achieve accuracy levels barely pass 73% accuracy[21]. This indicates that dataset characteristics and image quality play a crucial role in the overall accuracy achieved by the models.

One of the highest achieved accuracy with VGG architecture with Fer2013 is 73.28%[17]. VGG architecture interpreted in their own variant and data augmentation section includes randomly made ± 20% rescaling, 20% horizontal and vertical ± shift, and 10-degree rotation. Training is preferred instead of fine-tuning and training the last 300 epochs. Standard gradient descent with a 0.001 learning rate is used as an optimizer.

Another research that obtained an accuracy very close to the best result in VGG architecture has 72.7% accuracy[27]. The best accuracy result of fine-tuning the VGG16 model with fer2013 is currently 69.40%[19].

Another research used the Fer2013 dataset for the face expression recognition task. VGG16 and ResNet50 deep CNN architectures were used to make classification[18]. The different step in this study is that VGG16 and ResNet50 models are pre-trained with the VGGFACE dataset, which is currently unavailable. The results obtained by pre-trained with VGGFACE and fine-tuned with Fer2013 datasets are as follows; validation accuracy of VGG16 70.20% and ResNet50 73.20%.

The VGG16 model is widely used in this field, while the more recent variant of VGG, VGG19, is rarely used. Some studies implement the VGG19 architecture as one of the sub-networks within their own model and show achieved accuracy of 62.31% only for the VGG19 model[15]. According to Kusuma et al.(2020), the selection of learning layers depends on the model and dataset used. The objective of the work was

to establish a foundation for the VGG16 model. The final decision was that the best choice of learning layer is global average pooling. Similar processes are implemented in this study.

Another CNN architecture that is used in this study is Inception. Inception is used in various areas and established excellent performance. This architecture is one of the architectures that have state-of-art study in literature. When the Inception model is fine-tuned with the Fer2013 dataset, it showed remarkable performance in accuracy with 71.60%[27]. The latest version was introduced in 2016 and showed its performance in ImageNet Large Scale Visual Recognition Challenge(ILSVRC) 2015[38]. Another noteworthy study by Meena et al. achieved an accuracy of 73.09% using the InceptionV3 architecture[25]. The difference between the state-of-the-art study and this thesis is that a flattened layer was added to the model. However, detailed information regarding data augmentation and pre-processing processes was not extensively provided in the study.

In their experiment, Meena et al. used 224x224 image resolution and maintained the same size of batches for training, testing, and validating, which included ten samples each. Additionally, the callback function is used to store and reuse the model with the lowest validation loss.

Residual network architectures have demonstrated excellent accuracy across various computer vision tasks like object detection, image classification, segmentation, and more. It performed better accuracy than all the previous models in the literature. All of this leads to residual networks being the benchmark for evaluating the performance of new models[14]. The latest version of the Residual Network architecture is ResNet152V2. Notably, this model showed impressive results from various computer vision tasks. Before this thesis experiments, this model had never been used in face expression recognition tasks, especially with Fer2013.

5.3 CNN models trained with FACES dataset

There are fewer studies conducted using the FACES dataset[8] compared to the Fer2013 dataset. The FACES dataset is not publicly accessible, limiting its availability for researchers. The FACES dataset has a balanced structure since it contains an equal

number of each facial expression. It is a much smaller dataset than Fer2013 and contains posed images. The FACES contains six facial expressions with a total of 2052 images. All images were captured in a controlled laboratory environment with the same lightning conditions and resolution.

Among the studies conducted on the FACES dataset, one study performed two sets of experiments. The first set involved face expression recognition using classic CNN architectures, while the second set utilized deep CNN architectures such as InceptionV3, VGG19, and VGGFACE[32]. In this study, the validation accuracy obtained with InceptionV3 is 82.19%, and VGG19 is 97.19%. It is important to note that these validation accuracy values are obtained from the dataset where the training, validation, and test set could potentially contain images of the same people.

In another study, a model comparison was conducted only on elderly individuals from the FACES dataset[3]. The older versions of the models used in this thesis, VGG16, and InceptionV1/GoogleNet, were compared. Only fully connected layers were used as top layers. The study achieved a validation accuracy of 97.21% with VGG16 combined with a random forest classifier. In this study, the model was exposed to the same individuals in both the validation and training sets.
These are the performance results and approaches observed in published studies on the FACES dataset.

CHAPTER 6

METHODOLOGY AND RESULTS

In this chapter, the methodology and results of the explanatory comparative study of deep learning architectures will be explained in detail. Evaluation measures in computer vision tasks vary according to the specialty of the task. Selected evaluation measures the one that would be the most suitable for the FER task. Test accuracy results are generally used as evaluation measurements in this field. The evaluation measurement used can be explained as follows. Accuracy provides an overall evaluation of model performance when all classes have equal importance. The accuracy result is calculated by dividing the number of correct predictions by the total number of predictions made. Validation accuracy estimates how well the model classifies unseen data in the validation set after training. Both used datasets in this thesis are separated into three parts; 60% training data, 20% validation data, and 20% test data. Since the datasets are divided into three parts like this, the model validation and test are based on the images it has not seen before.

6.1 Training Strategies

Training strategies are a critical part of the process in face expression recognition tasks. They are a collection of techniques to optimize model performance in the training phase. The training process includes modifying the model's hyperparameters based on provided training data to increase the generalization capacity of the model. After this process, the model can make more accurate predictions on unseen images. Selecting proper training strategies is critical to achieving optimal performance in face expression recognition. These strategies are essential in enhancing

model convergence, handling limited training data, avoiding overfitting, optimizing hyperparameters, and managing imbalanced data.

6.1.1 Image pre-processing

Pre-processing techniques are essential for developing the quality of the input data and improvement of the performance of the model. The improvement of the FER models is developed by using multiple techniques, including image resizing, normalization, and data augmentation. This section provides an analysis of applied pre-processing techniques and their impact on training data. In all experiments, the same data pre-processings are applied to ensure that all models are comparable.

6.1.1.1 Image Re-sizing

The image resizing process includes modifying the dimensions of the facial expression images to conform to a standardized size. This process is fundamental to ensure consistency in image resolution, enabling the training process. All input images in the model should be the same size and have the same size as the input size of the model. Resizing the images in the dataset allows the model to consistently focus on relevant facial features for better learning and generalization of patterns associated with facial expressions[34]. While the dimensions of the images in Fer2013 are 48x48, they are resized to 224x224 to fit the input shape of the models. When image resizing using the Keras library, the default approach is set as the nearest approach. In the nearest approach, each pixel in an image is replaced based on the closest pixel in the original image.

6.1.1.2 Normalization

The normalization process is standardizing the values of the pixels in the image. The process includes multiple techniques like standard deviation scaling, min-max scaling, and z-score standardization. Normalization is applied to ensure that pixel intensities through different images can be compared and show consistency. Including this

Figure 6.1: Neutral Image Example Resized 48x48 to 224x224

process is critical to minimize the impact of lighting conditions, color variations, and contrast variations through the images. It helps to improve the ability of the model to generalize its performance across a wide range of images.
In this thesis, normalization is applied through the 'rescale=1./255' parameter. It scales down the pixel values in an image to between 0 and 1. Indicated rescale means dividing each pixel value by 255. The maximum pixel value in an 8-bit grayscale image is 255. Through this, the model's training process becomes more numerically stable and can be effective for gradient-based optimization.

6.1.1.3 Data Augmentation

Data augmentation is a commonly used technique in FER tasks. This process is applied to increase the variety and size of the training dataset. The process includes the application of multiple transformations to the input images, such as rotations, flips, and changing the contrast. Data augmentation helps to improve the model's ability to identify changes in facial poses, lighting conditions, and other factors. Furthermore, expanding the training data helps deal with the limited training data.
The following data augmentations are applied; rotation range of is 20%, width shift of 10%, height shift of 10%, horizontal flip, and zoom with a range of 20%. Since the facial image data is used, adding a vertical flip in the data augmentation process would cause serious degradation in the learning process. All data augmentation processes are only applied to the training sets in both datasets, and no data augmentation steps are applied to the validation and test sets.
In all experiments, the same data augmentations are applied to the images.

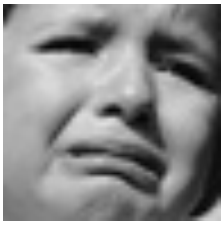


Figure 6.2: Original Image

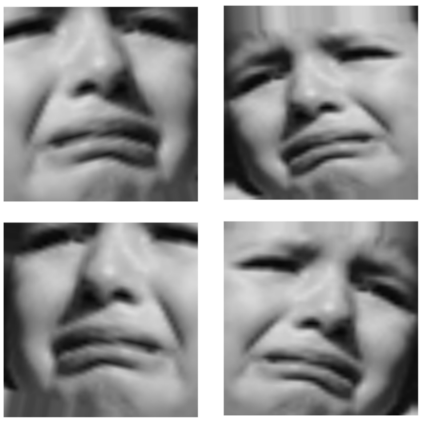


Figure 6.3: After applying the data augmentation techniques, 4 examples of the original image

6.1.2 Model Initialization and Optimization Algorithms

In the CNN architectures, multiple strategies exist to initialize the network parameters. These strategies involve random initialization, pre-training on large datasets, and using the pre-trained models.

This section details which model initialization parameters are used and pre-train models. Implementing pre-trained models means implementing existing well-performing CNN models that have been trained on similar tasks or a large dataset. Pre-trained models are also known as backbone models. These models are trained on large datasets to learn comprehensive and generalized features. Using pre-trained models can be more beneficial than training from scratch because of the already learned representations. It is practical to use pre-trained models when there is limited training data. In this thesis, pre-trained models are used with ImageNet weights in most of the experiments due to limited datasets.

As an optimization algorithm, standard gradient descent is implemented. After many trials, the SGD optimization algorithm is preferred as the most suitable algorithm for the FER task. As the learning rate determines the step size of the SGD, selecting the most suitable learning rate is fundamental. A well-selected learning rate is crucial to avoid convergence issues. The learning rate is set to 0.001 in SGD. This value for the learning rate is in the range of mostly used learning rate values. This value is selected after multiple training experiments.

The momentum of the SGD is selected as 0.9. The last parameter set in the SGD is Nesterov acceleration. Nesterov acceleration is used in the SGD. This enables gradient calculation updates to become more accurate and converge faster.
Instead of taking the top layers of the backbone models, custom top layers are added to the models. The experiments showed that the best fit for the top layers is global average pooling, followed by the Dense layer. The GAP layer helps to collect the information across the feature maps and capture global patterns on FER, which increases generalization capabilities. Adding the Dense layer after the GAP layer introduced non-linear and enabled learning of complex patterns. Using both layers together affects the learning of global and local features in FER. Finally, the softmax layer was added for classification.

6.1.3 Learning Rate Scheduling and Regularization

Learning rate scheduling is used to improve the training stability and generalization in the model's performance. As a learning rate scheduler, ' ReduceLROnPlateu' is implemented. The learning rate of the model is decreased when a monitored metric, in this case, the validation accuracy, reaches a plateau. This adjustment helps to avoid model overshooting and provides a stable model training process. The parameters selected in the learning rate scheduler are factor set to 0.2, patience to 5, and 'min lr' to 0.0001. Patience is that if the model does not improve its validation accuracy for five epochs, then it decreases the learning rate.
As a regularization technique, early stopping is implemented to monitor the validation accuracy and stops the training process if the model does not show any further improvement after five epochs.

6.2 Experiments

This thesis conducts multiple experiments on the FER task to answer various research questions and present the outcomes. The experiments included a comprehensive assessment of 5 CNN architectures using the Fer2013 and FACES datasets, allowing an extensive analysis of their performance and capabilities in FER.

6.2.1 Effects of Transfer Learning

Experiments are conducted to show the effect of transfer learning on the FER task. The first experiment is training the models from scratch without using pre-trained weights. The second experiment includes transfer learning with models that are pre-trained on ImageNet.

The pre-trained models are then trained with the Fer2013 dataset. All layers are trainable from the start. The results obtained from this training show that the highest accuracy belongs to VGG19 with %71.60. The results of these trainings are shown in Table 6.1.

Another training setup is applied to CNN architectures. This time models are trained from scratch with the Fer2013 dataset without including the pre-training weights of ImageNet. The results obtained from these experiments can be found in Table 6.1.

Model	Pre-training Dataset	Training Accuracy	Validation Accuracy	Test Accuracy	F1 Score
VGG19	ImageNet	94.77%	71.60%	**71.67%**	0.65
	-	74.27%	64.41%	64.22%	0.62
InceptionV3	ImageNet	96.45%	71.20%	71.38%	0.65
	-	80.67%	67.90%	69.23%	0.60
ResNet152V2	ImageNet	93.35%	70.68%	71.38%	0.64
	-	71.70%	63.97%	**71.35%**	0.59
MobileNetV2	ImageNet	84.99%	68.87%	69.49%	0.65
	-	73.29%	67.14%	67.26%	0.62
EfficientNetV2B1	ImageNet	63.90%	61.13%	61.38%	0.50
	-	55.69%	55.14%	55.71%	0.43

Table 6.1: The results obtained by training with Fer2013 both pre-trained and non-pre-trained models using the ImageNet dataset.

Without transfer learning, the ResNet152V2 model achieved the highest test accuracy with 71.35%. After the ResNet152V2 model, the second best results were obtained from InceptionV3 with 69.23%, and there is less than 1.0% between them. It is wrong

to say that one model is the best of them. After the top two models, MobileNetV2 comes with 67.26% test accuracy, VGG19 with 64.22%, and EfficientNetV2B1 with 55.71%.

The models trained with transfer learning using ImageNet showed higher test and training accuracy, as shown in the table. On the other hand, the models trained without transfer learning showed different levels of validation and training accuracy values. The findings show that implementing transfer learning with ImageNet weights improves the model's performance.

Khanzada et al. conducted a study using VGG16 and ResNet50 models pre-trained on the VGG-Face dataset with the Fer2013 dataset[18]. The pre-trained weights used in that study belong to the VGG-Face dataset of Oxford University[26]. The state-of-art study that used VGG CNN architecture in face expression recognition also applied a pre-training process with a different dataset than Imagenet.
In order to examine the effect of the pre-training dataset, training was performed with identical versions of models. The results obtained from this training can be found in Table 6.2. VGG16 and ResNet50 CNN architectures were modified with the same top layers as the study, global average pooling. Both of the models were trained with three different configurations in the pre-training dataset process. The first training was conducted without any transfer learning. The second training was conducted after the pre-training was applied with the VGG-Face dataset. The last one was conducted with the most popular pre-training dataset, the ImageNet dataset. After these six different training processes were applied, the results in Table 6.2 were obtained. After looking at the results, it can be said that pre-training with the dataset containing facial expressions has a positive effect on the results.

Model	Pre-Traning Dataset	Trained Dataset	Input Size	Training Accuracy	Validation Accuracy	Test Accuracy
VGG16	-	Fer2013	(224,224,3)	82.92%	67.59%	64.50%
	VGGFace	Fer2013	(197,197,3)	92.78%	71.27%	71.44%
	ImageNet	Fer2013	(224,224,3)	95.31%	69.37%	70.74%
ResNet50	-	Fer2013	(224,224,3)	67.91%	63.24%	70.66%
	VGGFace	Fer2013	(197,197,3)	97.66%	73.19%	73.83%
	ImageNet	Fer2013	(224,224,3)	97.03%	70.01%	70.77%

Table 6.2: The VGG16 and ResNet50 models were trained from scratch, pre-trained with VGGFace, and pre-trained with ImageNet, and subsequently, all models were trained and tested using the Fer2013 dataset.

These experiments provide valuable insights into the advantages and disadvantages related to transfer learning. When the results shown in Table 6.2 are examined, it can be said that there is a difference between the pre-trained models with the VGGFace dataset and the pre-trained models with ImageNet. VGGFace dataset is not used for three main reasons. The use of this dataset is less common in the literature, which is a significant obstacle to direct comparison with other studies. Another reason is that the time required to train models from scratch is too long. VGGFace dataset is not publicly accessible. After getting these results, all subsequent training is conducted on models pre-trained with ImageNet.

6.2.2 Comparative Results on Fer2013

Experiments are started by fine-tuning VGG19, InceptionV3, ResNet152V2, MobileNetV2, and EfficientNetV2B1 with Fer2013 dataset. While fine-tuning the models, pre-training weights from the ImageNet dataset are used. While fine-tuning the models, all of the layers are set as trainable. The inceptionV3 model has the longest training time, while the EfficientNetV2B1 model has the shortest training time. Validation and training accuracy graphs belonging to these experiments are shown in Figure 6.4. The results obtained from these experiments with Fer2013 are shown in Table 6.4. By looking at this table, it can be seen that the test accuracy values are mostly close to each other, with the exception of EfficientNetV2B1, but the best result belongs to VGG19.

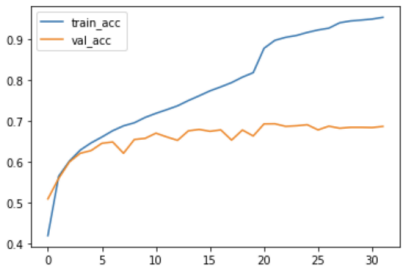
Model	Training Accuracy	Validation Accuracy	Test Accuracy
VGG19	94.77%	71.60%	**71.67%**
InceptionV3	96.45%	71.20%	71.38%
ResNet152V2	93.35%	70.68%	71.38%
MobileNetV2	84.99%	68.87%	69.49%
EfficientNetV2B1	63.90%	61.13%	61.38%

Table 6.3: The results obtained from training and testing the models with Fer2013 dataset.

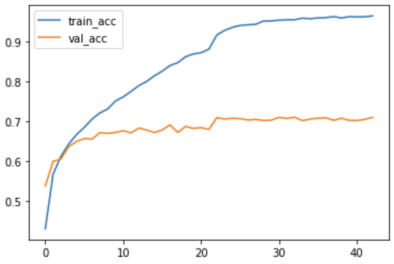
This thesis uses optimal top layers that can be used with various CNN architectures, enabling a fair comparison across different models. Therefore, surpassing the state-of-art accuracy for the InceptionV3 model was not accomplished. In other words, the performance of the proposed approach did not outperform the current highest accuracy achieved by the InceptionV3 model.

Paper	Model	Transfer Learning	Test Accuracy
Khaireddin2021	Custom VGG	-	**73.28%**
Kusuma2020	VGG16	+	69.40%
Pramerdorfer2016	VGG	-	72.70%
	Inception	-	71.60%
	ResNet50	-	72.40%
Our Study	VGG16	+	70.30%
	VGG19	+	**71.60%**
	InceptionV3	+	71.02%
	ResNet152V2	+	70.68%
	MobileNetV2	+	68.87%
	EfficientNetV2B1	+	61.13%

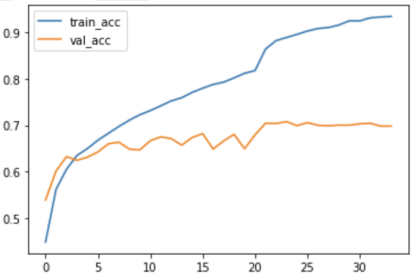
Table 6.4: Comparison of Test Accuracy: State-of-the-Art Studies vs. Our Results



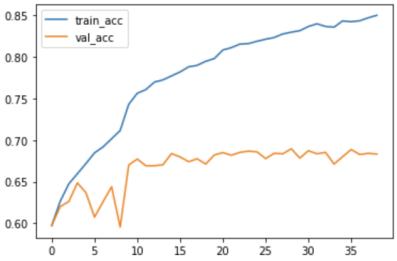
(a) VGG19 validation and training accuracy graph



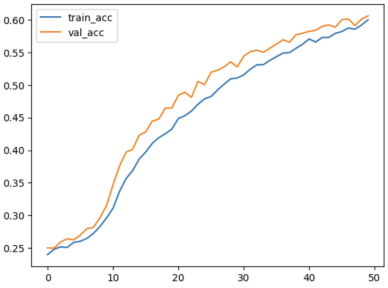
(b) InceptionV3 validation and training accuracy graph



(c) ResNet152V2 validation and training accuracy graph



(d) MobileNetV2 validation and training accuracy graph



(e) EfficientNetV2B1 validation and training accuracy graph

Figure 6.4: Models fine-tuned with Fer2013 dataset

The accuracy values obtained by testing the models fine-tuned with Fer2013 on the FACES datasets are shown in Table 6.5. The FACES dataset was used as a separate test set to test the generalization capacity of models fine-tuned on Fer2013. The question we want to answer is how these models fine-tuned on Fer2013 perform on images that are significantly different from the ones used in training.

The test accuracy values on the FACES dataset are lower than those obtained from Fer2013, as expected (Table 6.5). This difference in test accuracy results can be associated with multiple factors. Test accuracy results are affected by quality, size, and data distribution on the dataset. When the model is trained on a dataset, it becomes familiar with its features and patterns. As a result of this, when the testing is applied with the test set from the same dataset, they show high test accuracy results. However, when the models are tested with the FACES dataset, test accuracy results have lower values than before. After looking at these results, information about the generalization capacity between the models can be obtained. Looking at the results, the big difference between VGG19 and EfficientNetV2B1 stands out. This difference is actually based on their generalization capacity. While VGG19 has the highest generalization capacity, we can reach the information that EfficientNetV2B1 only memorizes.

Model	Training Accuracy	Validation Accuracy	Test Accuracy with Fer2013	Test Accuracy with FACES
VGG19	94.7%	71.60%	**71.67%**	**59.84%**
InceptionV3	96.45%	71.20%	71.38%	52.40%
ResNet152V2	93.35%	70.68%	71.38%	49.31%
MobileNetV2	84.99%	68.87%	69.49%	34.51%
EfficientNetV2B1	63.90%	61.13%	61.38%	10.18%

Table 6.5: All models fine-tuned with Fer2013 test accuracy from Fer2013 and test accuracy from FACES

6.2.3 Comparative Results on FACES

Experiments are conducted by fine-tuning with the FACES dataset on VGG19, InceptionV3, ResNet152V2, MobileNetV2, and EfficientNetV2B1. All layers of the models were set as trainable in this experiment. In order to be able to make a comparison with the studies in the literature, all of the models are fine-tuned with two

different approaches.

Literature studies that used the FACES dataset for face expression recognition tasks are reviewed. In the studies, it is not taken into consideration that the images belonging to the same people are included in all of the training, validation, and test set. Since the FACES dataset consists of the same people, two different experiments are conducted on the model.

In the first experiment, we ensured that the validation, training, and test sets consisted of face expressions from different individuals. This way, each set contained distinct people, reducing the potential for bias and enabling a fair evaluation of the model's performance. The data separation method used in this experiment was called manual split.

For the second experiment, we took specific precautions during the separation of the datasets. While dividing the data for training with face images, we made sure that individuals who appeared in the training set were not present in the validation set. This approach helped prevent any overlap of data and ensured that the model was trained and evaluated on different individuals for each phase. The data separation method used in this experiment was called random split.

By conducting these two experiments, we aimed to mitigate any potential biases and ensure a robust evaluation of the model's performance on the FACES dataset for face expression recognition. This approach allows for a more accurate comparison of the model's results with other literature methods and enhances our findings' reliability.

The results obtained from these experiments with the FACES dataset are shown in Table 6.6. As can be seen from the table, the best test accuracy values are obtained in both experiments, with the highest result in VGG19.

	Manual Split				**Random Split**			
Model	Training Accuracy	Validation Accuracy	Test Accuracy	F1 Score	Training Accuracy	Validation Accuracy	Test Accuracy	F1 Score
VGG19	98.03%	92.13%	**94.44%**	0.94	99.81%	99.27%	**98.87%**	0.97
InceptionV3	99.92%	94.59%	93.51%	0.93	99.87%	98.29%	98.32%	0.97
ResNet152V2	99.47%	94.84%	92.28%	0.93	99.63%	97.59%	97.68%	0.96
MobileNetV2	91.44%	80.57%	82.71%	0.83	99.57%	97.81%	97.88%	0.97
EfficientNetV2B1	88.7%	82.30%	84.56%	0.85	99.87%	98.29%	98.33%	0.98

Table 6.6: Training results obtained from manually and randomly splitting the FACES dataset.

In order to make a comparison analysis with the results presented in this thesis and state-of-the-art, it is necessary to refer to Table 6.7. In this table, a comparison is made with the results obtained from the random split. After looking at the table, it can be seen that the values obtained in the results comparable to the state of art studies surpass their results. The reason for this may be the using different learning layers or the effect of the steps applied in data pre-processing.

Paper	Model	Transfer Learning	Test Accuracy
Sajjanhar2018	VGG19	+	97.16%
	InceptionV3	+	82.19%
Our Study	VGG19	+	**98.87%**
	InceptionV3	+	98.32%

Table 6.7: Comparison of Test Accuracy: State-of-the-Art Studies vs. Our Results

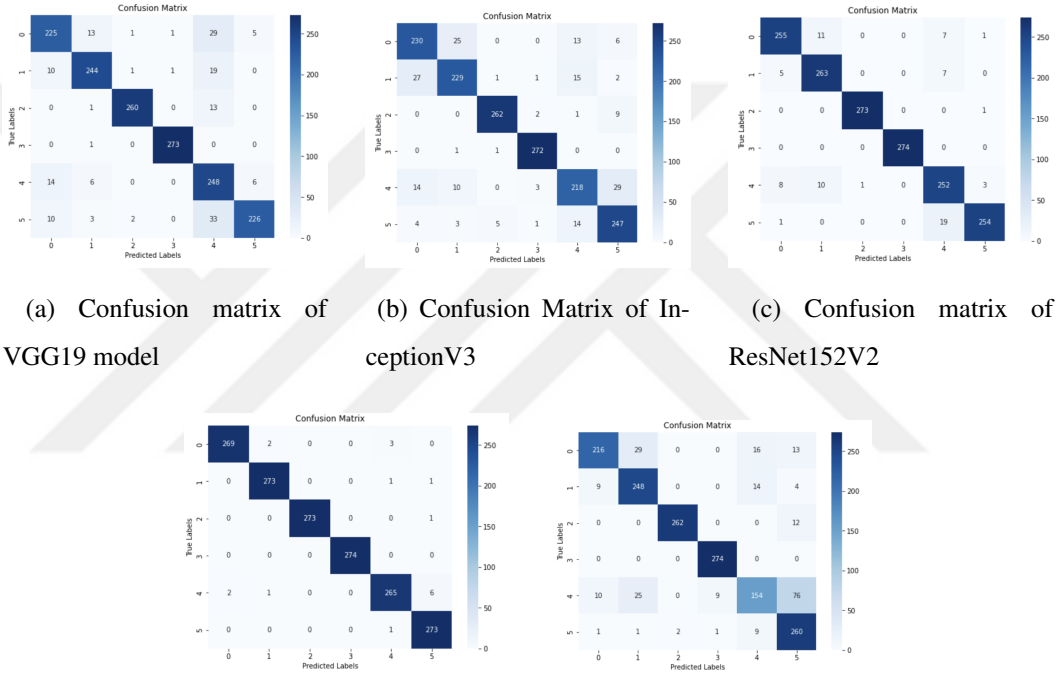
6.2.4 Effects of Fer2013 Pre-training on FACES dataset

In the 6.2.3 section, models were initialized with ImageNet weights and trained with the FACES dataset. The experiments aimed to see the impact of pre-training on the fer2013 dataset on test performance with the FACES dataset. To achieve these, the models were initially pre-trained on the Fer2013 dataset. Subsequently, fine-tuning of these pre-trained models is conducted, followed by their evaluation of the FACES dataset with a manual split. Instead of making all layers trainable in this experiment, only learning layers are set as trainable. In the other experiment, all models are all layers fine-tuned with FACES in Table 6.8.

	Fine-tuning of Only Top Layers				**Fine-tuning of All Layers**			
Model	Training Accuracy	Validation Accuracy	Test Accuracy	F1 Score	Training Accuracy	Validation Accuracy	Test Accuracy	F1 Score
VGG19	89.72%	81.81%	89.73%	0.90	99.81%	99.27%	92.28%	0.92
InceptionV3	88.63%	79.11%	91.40%	0.91	99.87%	98.29%	91.35%	0.98
ResNet152V2	95.50%	84.76%	95.50%	0.96	99.63%	97.59%	**92.59%**	0.96
MobileNetV2	98.90%	81.81%	**98.91%**	0.99	99.57%	97.81%	87.03%	0.97
EfficientNetV2B1	85.95%	82.80%	85.96%	0.86	99.87 %	98.29%	90.12%	0.86

Table 6.8: Comparison of fine-tuning of only top layers and fine-tuning of all layers with the FACES dataset

The following confusion matrices show prediction values for each expression with 5 CNN architectures for the following confusion matrices after the fine-tuning process with the FACES dataset; it can be said by looking at these confusion matrices which expression has the best accuracy percentage than other expressions. The classes and what these classes represent are shown in Table 6.9. When each expression is examined, happy is more accurate than all other expressions. This is due to the fact that there are too many images in the happy class in Fer2013 dataset. When looking at which class has the worst accuracy, except for VGG19, sad is the most challenging expression to classify for the other four CNN architectures. For the VGG19 model, this expression is anger with 82.12%.



(a) Confusion matrix of VGG19 model

(b) Confusion Matrix of InceptionV3

(c) Confusion matrix of ResNet152V2

(d) Confusion Matrix of MobileNetV2

(e) Confusion Matrix of EfficientNetV2B1

Figure 6.5: Confusion matrices of models trained with Fer2013, fine-tuned and tested with FACES dataset

Class Number	0	1	2	3	4	5
Expression	Anger	Disgust	Fear	Happy	Sad	Neutral

Table 6.9: Expression and Corresponding Number

```
Class 0: Accuracy = 82.12%
Class 1: Accuracy = 88.73%
Class 2: Accuracy = 94.89%
Class 3: Accuracy = 99.64%
Class 4: Accuracy = 90.51%
Class 5: Accuracy = 82.48%
```

(a) Percentage for each class - VGG19

```
Class 0: Accuracy = 83.94%
Class 1: Accuracy = 83.27%
Class 2: Accuracy = 95.62%
Class 3: Accuracy = 99.27%
Class 4: Accuracy = 79.56%
Class 5: Accuracy = 90.15%
```

(b) Percentage for each class - InceptionV3

```
Class 0: Accuracy = 93.07%
Class 1: Accuracy = 95.64%
Class 2: Accuracy = 99.64%
Class 3: Accuracy = 100.00%
Class 4: Accuracy = 91.97%
Class 5: Accuracy = 92.70%
```

(c) Percentage for each class - ResNet152V2

```
Class 0: Accuracy = 98.18%
Class 1: Accuracy = 99.27%
Class 2: Accuracy = 99.64%
Class 3: Accuracy = 100.00%
Class 4: Accuracy = 96.72%
Class 5: Accuracy = 99.64%
```

(d) Percentage for each class - MobileNetV2

```
Class 0: Accuracy = 78.83%
Class 1: Accuracy = 90.18%
Class 2: Accuracy = 95.62%
Class 3: Accuracy = 100.00%
Class 4: Accuracy = 56.20%
Class 5: Accuracy = 94.89%
```

(e) Percentage for each class - EfficientNetV2B1

Figure 6.6: Percentage of correctly labeled images for each class for each model trained with Fer2013 and fine-tuned and tested with Faces

Compared to all layers fine-tuned results, there is an increase in the test accuracy from the top layers fine-tuned. When looking at the test accuracy results, some models did not show improvement. While VGG19 and EfficientNetV2B1 achieve higher test accuracies, InceptionV3, MobileNetV2, and ResNet152V2 model's test accuracy values decrease. When all the layers are set the trainable, the highest test accuracy is obtained from the ResNet152V2 model with 92.59%. On the other hand, the highest test accuracy obtained from experiments is 98.91% with MobileNetV2, fine-tuned with only top layers.

6.2.5 Effects of Progressive Fine-tuning

In this thesis, Progressive is defined as fine-tuning as a training technique that first fine-tunes the specific number of layers in the model and then fine-tunes the entire model. This technique allows the model to progressively adapt to the given task with the knowledge gained from previous training. The progressive learning technique is shown in Figure 6.7.

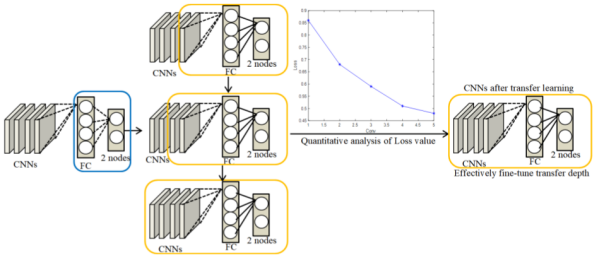


Figure 6.7: Progressive fine-tuning technique[40]

VGG19 model is used to examine the effects of fine-tuning all layers at once and progressive fine-tuning on the results. Firstly only the top layers are unlocked. After the training of the top layers is completed, the rest of the model is unlocked and fine-tuned. Results are shown in Table 6.10.

	Training Accuracy	Validation Accuracy	Test Accuracy
Fine-tuned with all layers	92.45%	70.01%	**70.38%**
Progressive Fine-tuning	82.24%	68.89%	**69.29%**

Table 6.10: Compare the results of fine-tuning the VGG19 model with all layers unlocked versus progressive fine-tuning.

Progressive fine-tuning does not ensure higher accuracy than full fine-tuning. In contrast to the findings reported in the literature, experiments did not yield any improvement in the obtained results. The implementation of progressive fine-tuning in facial expression studies may not yield significant benefits and could be considered inefficient.

CHAPTER 7

DISCUSSION AND CONCLUSION

In this thesis, it is aimed to provide an explanatory comparative study of five popular and different kinds of CNN architectures, which are listed as VGG19, InceptionV3, ResNet152V2, MobileNetV2, and EfficientNetV2B1, for face expression recognition task. The same data pre-processing and hyperparameters are used on all CNN architectures. The same data augmentation steps were applied to the images in both datasets used in this thesis. All of the parameters affecting the training process, including the pre-processing, data augmentation, and optimization parameters, were selected by hyperparameter tuning. As a result of these experiments, the parameters that give optimal validation accuracy in all models were implemented.

Multiple experiments are conducted to observe the effects of transfer learning with 5 CNN architectures. Experiment details are explained in the section 6.2.1, and results are shown in Table 6.1. When training durations are examined, it is observed that transfer learning leads to much faster training. Training durations are shown in Figure 7.1 The results showed that transfer learning performed consistently better than training from scratch. The VGG19 model has the highest accuracy, while the EfficientNetV2B1 model has the lowest accuracy. These results showed the effectiveness of using the pre-trained models. Transfer learning should be used to improve the model's performance on FER tasks.

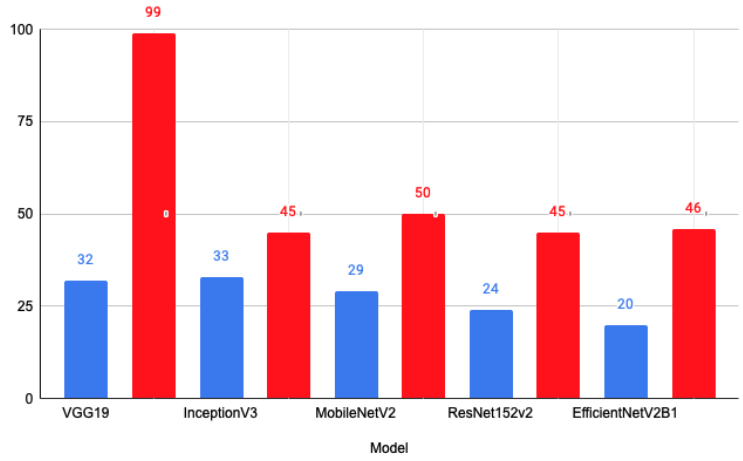


Figure 7.1: Training duration of 5 models with transfer learning(blue) and without transfer learning(red).

When the FACES dataset is split randomly, the best accuracy is performed by the VGG19 model with 99.27% validation accuracy and 98.87% test accuracy. When the FACES dataset is split according to people, the best accuracy was performed by ResNet152V2 with 94.84% validation accuracy, and the best test accuracy was performed with VGG19 94.44%. Although the ResNet152V2 model achieved the best validation accuracy, the model achieved the third-best test accuracy values with 92.28%. When the training results obtained with VGG19 are examined, it can be seen that the model achieves better results than other models. The major difference between VGG19 and the other models is that VGG19 has much more parameters. Table 3.1 shows which CNN architecture has how many parameters. The more the number of trainable parameters increases, the more likely that the model gets better results[35].

However, increasing the number of parameters is not always a solution because it has many negative effects, such as the long duration of the training process. Some of the challenges of working with a high number of parameter models are that a high number of parameters means higher processing loads, high memory access, and high energy consumption. These negative effects make the model hard to adapt in compact devices[36]. These effects should be considered while choosing a CNN architecture for the project.

One of the research questions in this thesis is what are the outcomes from the training and fine-tuning with unbalanced datasets such as Fer2013 and a posed and balanced dataset such as FACES. In order to make this analysis, the models trained with the Fer2013 dataset are fine-tuned with the FACES dataset and tested with the FACES test set. The results obtained from the training where only two learning layers and all layers are set to trainable are shown in table 6.8. An increase in accuracy is observed after the models are fine-tuned with the FACES dataset. Figure 7.2 shows the accuracy results obtained from this training. When the validation accuracy results are observed from Table 6.8, it can be seen that the best improvement is performed with EfficientNetV2B1. The EfficientNetV2B1 model is the most open model to improve through these characteristics in its architecture; scalability, newest improvements in the architecture, and good with data diversity[39]. When looking at the results in terms of test accuracy and validation accuracy overall, it can be said that the ResNet152V2 model is the most successful model in two layers of fine-tuning. Then looking at the overall performance, as said before, ResNet152V2 performed more constant and more reliable improvement results.

These results show the model's ability to transfer knowledge from pre-training with Fer2013, fine-tuning with the FACES dataset. Fine-tuning with learning layers or all layers affects each model's performance differently. This indicates the importance of choosing the right fine-tuning strategies depending on the task.

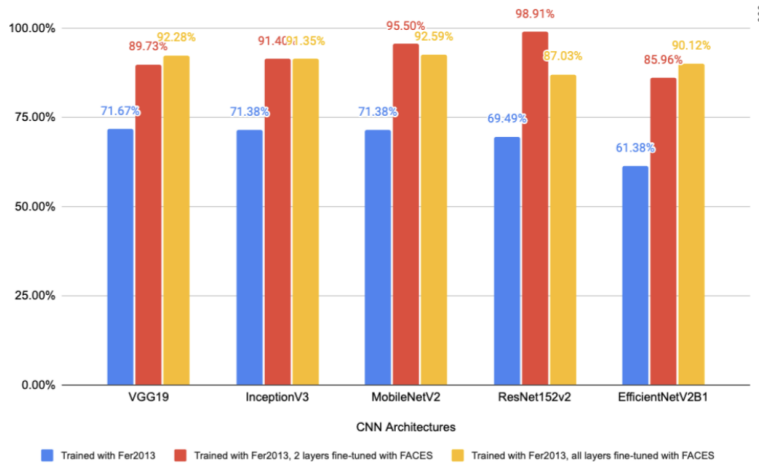


Figure 7.2: Test Accuracy on FACES from only training with Fer2013, training with Fer2013 , fine-tuned with top layers with Faces and trained with Fer2013, all layers fine-tuned with Faces

Based on these results, if training will be performed with large and unbalanced datasets, it is better to train with deep but simpler CNN architecture such as VGG19. Other model ResNet152V2 should be preferred if training with a large and unbalanced dataset is proposed, which is the technique usually applied while training and then fine-tuning with a balanced and posed dataset. If there is no concern for energy consumption or memory space while training, VGG19 models should be selected thus that all of the layers are set to trainable. These statements are valid if the models will not be integrated into an embedded system. If it is integrated into an embedded system, the MobileNetV2 model should be selected as it showed the highest test accuracy.

In conclusion, the results obtained from the experiments provided detailed information about the strengths and weaknesses of all five models in face expression recognition. Each of the five CNN architectures showed their advantages and disadvantages via these experiments.

VGG19 architecture exhibited its depths and ease of implementation advantages by getting the highest accuracy from Fer2013 training. On the other hand, ResNet152V2 showed its quickness and excellence in complex feature capture by getting the highest accuracy from fine-tuning with the FACES dataset. InceptionV3 showed that it's the fastest and the best adapted to the new dataset without using the pre-training weights. MobileNetV2 model provides lightweight, suitable models for the applications. EfficientNetV2B1 demonstrated its efficiency and scalability when trainable parameters increase in the training process with getting the highest accuracy from FACES all layers trainable fine-tuning.

In addition, it is observed how much the characteristics of the facial datasets, such as the size and quantity distribution according to each class, affect the training.

Although this thesis has provided detailed information and insights into the performance of five different kinds of CNN architectures in face expression recognition, there are still multiple potential developments for future studies. If the focus is to achieve better accuracy results rather than comparing the architectures, then architectures can be modified more. Hybrid architectures can be used by getting the strengths of each architecture and combining them. For instance, combining the ResNet152V2 and EfficientNetV2B1 could lead to getting the advantageous features of both and getting better results. If future studies are not interested in making comparisons with

state-of-the-art or literature, then transfer learning and fine-tuning performances can be improved by expanding the facial dataset by adding different age groups, poses, and ethnicities. Likewise, if only a high accuracy value is desired, then the ensemble method can be applied. The ensemble method is, also called model averaging and model stacking, combines the prediction values of multiple models.

Real-time face expression recognition applications can be developed with the help of models trained in this thesis. This application can be on an edge device or mobile platform.

To summarise, the outcomes from the comparison of VGG19, ResNet152V2, InceptionV3, MobileNetV2, and EfficientNetV2B1 in face expression recognition provided in this thesis have given insight into how the model's performance and characteristics for different conditions. In future work, higher accuracy results can be performed by following the suggestions mentioned above and can be more focused on real-time applications. Data was split according to people. Previously referred to as manual split. We looked at how well the models classified people they had never seen in the test set. Afterward, the training step with a single photo can be implemented, and look at the test accuracy again. It can be observed if the change in results is for the better or worse. Did the model learn anything about the facial features of that person by seeing a single photo? Such types of questions may be answered in future studies.

REFERENCES

[1]Monika Bansal et al. “Transfer learning for image classification using VGG19: Caltech-101 image data set”. In: *Journal of ambient intelligence and humanized computing* (2021), pp. 1–12.

[2]Daniel Canedo and António JR Neves. “Facial expression recognition using computer vision: A systematic review”. In: *Applied Sciences* 9.21 (2019), p. 4678.

[3]Andrea Caroppo, Alessandro Leone, and Pietro Siciliano. “Comparison between deep learning models and traditional machine learning approaches for facial expression recognition in ageing adults”. In: *Journal of Computer Science and Technology* 35 (2020), pp. 1127–1146.

[4]Andrea Caroppo, Alessandro Leone, and Pietro Siciliano. “Facial expression recognition in ageing adults: A comparative study”. In: *Ambient Assisted Living: Italian Forum 2018 9*. Springer. 2019, pp. 349–359.

[5]François Chollet et al. “Keras: Deep learning library for theano and tensorflow”. In: *URL: https://keras. io/k* 7.8 (2015), T1.

[6]Roddy Cowie et al. “Emotion recognition in human-computer interaction”. In: *IEEE Signal processing magazine* 18.1 (2001), pp. 32–80.

[7]Shichuan Du and Aleix M Martinez. “Compound facial expressions of emotion: from basic research to clinical applications”. In: *Dialogues in clinical neuroscience* (2022).

[8]Natalie C Ebner, Michaela Riediger, and Ulman Lindenberger. “FACES—A database of facial expressions in young, middle-aged, and older women and men: Development and validation”. In: *Behavior research methods* 42 (2010), pp. 351–362.

[9]Paul Ekman. “An argument for basic emotions”. In: *Cognition & emotion* 6.3-4 (1992), pp. 169–200.

[10]Dario Garcia-Gasulla et al. “On the behavior of convolutional nets for feature extraction”. In: *Journal of Artificial Intelligence Research* 61 (2018), pp. 563–592.

[11]Ian J Goodfellow et al. “Challenges in representation learning: A report on three machine learning contests”. In: *Neural Information Processing: 20th International Conference, ICONIP 2013, Daegu, Korea, November 3-7, 2013. Proceedings, Part III 20*. Springer. 2013, pp. 117–124.

[12]Saad Hikmat Haji and Adnan Mohsin Abdulazeez. “Comparison of optimization techniques based on gradient descent algorithm: A review”. In: *PalArch’s Journal of Archaeology of Egypt/Egyptology* 18.4 (2021), pp. 2715–2743.

[13]Zabit Hameed et al. “Breast cancer histopathology image classification using an ensemble of deep learning models”. In: *Sensors* 20.16 (2020), p. 4373.

[14]Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[15]Wentao Hua et al. “HERO: Human emotions recognition for realizing intelligent Internet of Things”. In: *IEEE Access* 7 (2019), pp. 24321–24332.

[16]Mohammed Abbas Kadhim and Mohammed Hamzah Abed. “Convolutional neural network for satellite image classification”. In: *Intelligent Information and Database Systems: Recent Developments 11* (2020), pp. 165–178.

[17]Yousif Khaireddin and Zhuofa Chen. “Facial emotion recognition: State of the art performance on FER2013”. In: *arXiv preprint arXiv:2105.03588* (2021).

[18]Amil Khanzada, Charles Bai, and Ferhat Turker Celepcikay. “Facial expression recognition with deep learning”. In: *arXiv preprint arXiv:2004.11823* (2020).

[19]Gede Putra Kusuma, J Jonathan, and AP Lim. “Emotion recognition on fer-2013 face images using fine-tuned vgg-16”. In: *Advances in Science, Technology and Engineering Systems Journal* 5.6 (2020), pp. 315–322.

[20]Hanjiang Lai et al. “Deep recurrent regression for facial landmark detection”. In: *IEEE Transactions on Circuits and Systems for Video Technology* 28.5 (2016), pp. 1144–1157.

[21]Shan Li and Weihong Deng. “Deep facial expression recognition: A survey”. In: *IEEE transactions on affective computing* 13.3 (2020), pp. 1195–1215.

[22]Min Lin, Qiang Chen, and Shuicheng Yan. “Network in network”. In: *arXiv preprint arXiv:1312.4400* (2013).

[23]Muhammad Mateen et al. “Fundus image classification using VGG-19 architecture with PCA and SVD”. In: *Symmetry* 11.1 (2018), p. 1.

[24]Warren S McCulloch and Walter Pitts. “A logical calculus of the ideas immanent in nervous activity”. In: *The bulletin of mathematical biophysics* 5 (1943), pp. 115–133.

[25]Gaurav Meena, Krishna Kumar Mohbey, and Sunil Kumar. “Sentiment analysis on images using convolutional neural networks based Inception-V3 transfer learning approach”. In: *International Journal of Information Management Data Insights* 3.1 (2023), p. 100174.

[26]Omkar M. Parkhi, Andrea Vedaldi, and Andrew Zisserman. “Deep Face Recognition”. In: *British Machine Vision Conference*. 2015.

[27]Christopher Pramerdorfer and Martin Kampel. “Facial expression recognition using convolutional neural networks: state of the art”. In: *arXiv preprint arXiv:1612.02903* (2016).

[28]JR Rajayogi, G Manjunath, and G Shobha. “Indian food image classification with transfer learning”. In: *2019 4th International Conference on Computational Systems and Information Technology for Sustainable Solution (CSITSS)*. Vol. 4. IEEE. 2019, pp. 1–4.

[29]Viswanatha Reddy Gajjala et al. “MERANet: Facial Micro-Expression Recognition using 3D Residual Attention Network”. In: *arXiv e-prints* (2020), arXiv–2012.

[30]Frank Rosenblatt. “The perceptron: a probabilistic model for information storage and organization in the brain.” In: *Psychological review* 65.6 (1958), p. 386.

[31]David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. “Learning representations by back-propagating errors”. In: *nature* 323.6088 (1986), pp. 533–536.

[32]Atul Sajjanhar, ZhaoQi Wu, and Quan Wen. “Deep learning models for facial expression recognition”. In: *2018 digital image computing: Techniques and applications (dicta)*. IEEE. 2018, pp. 1–6.

[33]Mark Sandler et al. “Mobilenetv2: Inverted residuals and linear bottlenecks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 4510–4520.

[34]Florian Schroff, Dmitry Kalenichenko, and James Philbin. “Facenet: A unified embedding for face recognition and clustering”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 815–823.

[35]Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[36]Shahriar Shakir Sumit et al. “Restinet: On improving the performance of tiny-yolo-based cnn architecture for applications in human detection”. In: *Applied Sciences* 12.18 (2022), p. 9331.

[37]Christian Szegedy et al. “Going Deeper With Convolutions”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2015.

[38]Christian Szegedy et al. “Rethinking the inception architecture for computer vision”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2818–2826.

[39]Mingxing Tan and Quoc Le. “Efficientnetv2: Smaller models and faster training”. In: *International conference on machine learning*. PMLR. 2021, pp. 10096–10106.

[40]Zhen Wang and Shanwen Zhang. “Sonar image detection based on multi-scale multi-column convolution neural networks”. In: *IEEE Access* 7 (2019), pp. 160755–160767.

Appendix A

APPENDIX

The source code and additional resources related to this thesis can be found in GitHub account at the following link:

GitHub Repository: https://github.com/fyenilmez/TDU-Thesis

This repository is publicly accessible and contains the training codes, dataset pre-processing files, and additional Python files used in experiments.